Autonomous Visual Learning for Robotic Systems

submitted by

Daniel Beale

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Computer Science

March 2012

Summary

This thesis investigates the problem of visual learning using a robotic platform. Given a set of objects the robots task is to autonomously manipulate, observe, and learn. This allows the robot to recognise objects in a novel scene and pose, or separate them into distinct visual categories. The main focus of the work is in autonomously acquiring object models using robotic manipulation.

Autonomous learning is important for robotic systems. In the context of vision, it allows a robot to adapt to new and uncertain environments, updating its internal model of the world. It also reduces the amount of human supervision needed for building visual models. This leads to machines which can operate in environments with rich and complicated visual information, such as the home or industrial workspace; also, in environments which are potentially hazardous for humans.

The hypothesis claims that inducing robot motion on objects aids the learning process. It is shown that extra information from the robot sensors provides enough information to localise an object and distinguish it from the background. Also, that decisive planning allows the object to be separated and observed from a variety of different poses, giving a good foundation to build a robust classification model. Contributions include a new segmentation algorithm, a new classification model for object learning, and a method for allowing a robot to supervise its own learning in cluttered and dynamic environments.

Acknowledgements

Firstly, I would like to express my gratitude to my supervisors Peter Hall and Pejman Iravani. Their guidance, inspiration and support throughout my PhD will always be appreciated.

I would also like to thank all of my friends in the PhD lab, the friends I have made throughout my undergraduate degree, college and comprehensive school; their emotional support, encouragement and technical advice went a long way towards helping me to complete the thesis. There are far too many to mention here, but, you all know who you are!

The work was funded by the University of Bath which has been a home for me for many years and given me all the tools needed to build my career. Many thanks to the University, and all of the lecturers, staff and students within it, who have helped me to achieve my goals while providing support and guidance along the way.

Finally, I would like to thank my parents and family for their friendship, financial support and unconditional love throughout my life, without them, nothing would be possible.

Contents

Part I

Introduction

Chapter 1

Introduction

The holy grail of robotics is a machine that can operate as well as a human, but with the advantage of coping in environments that are otherwise inhospitable or too physically demanding. The world we live in is vastly complex, but despite that, we are able to operate in it with considerable ease. Millions of years of evolution has a part to play in the model we have of our universe, but in order for any intelligent agent to function properly it must first be able to adapt to its environment, learning in the presence of change. This thesis explores what it means for a robot to learn about objects in the environment, developing a learning framework for a robot to build robust object models.

Fundamental to cognition is the problem of sensing, for a robot to make decisions it must first be able to obtain information about its environment. One of the most important and information rich sensors is the vision system, allowing entities to observe and learn about objects in the world. The field of computer vision is vast, with work dating back to the 60's. The first people to study the problem were a group of psychologists, pioneered by [Marr, 1982], who suggested ideas for how the human brain represents visual data. Although the work remains fundamental to current vision algorithms, more recent work focuses on probabilistic methods to build visual models.

Current computer vision algorithms focus on the ability to learn, without assuming any control over the environment. In the context of robots with vision, the robot can either navigate the environment or move objects within it. The ability to plan a motion, and execute it, in order to actively change the environment, gives a robot more information to learn from than a passive vision

system.

When using a robot there is much more information available, it has a degree of control over the environment, the ability to choose to manipulate objects in a particular way. The ability to act on the environment conforms with another idea that evolved at around the same time as Marr. [Gibson, 1986] argued that objects are perceived by the actions that can be performed on them, by their 'affordances'. This idea explicitly links action with perception, something that has resonated throughout the robotics literature [Brooks, 1999] and more currently computer vision [Soatto, 2009].

The work presented in the thesis uses knowledge of the robot's motion, or in other words *proprioception*, to aid the perception and learning process. In other words, the robot is able to learn about objects placed in its workspace by manipulating them.

1.1 Research Hypothesis

The results presented in this thesis makes significant headway in proving the hypothesis that:

- **'Robot-object interaction, in the form of induced motion on objects, aids robot perception and the learning of visual object models.'**

The definition given above is a compact representation of the work presented in the thesis, and composed of terms explained in further detail below:

- The word 'perception' is defined as object level segmentation, using only robotic sensor data, and no prior knowledge.

- The term 'object level segmentation' means to take the image that the robot is viewing and remove the background, leaving only the object that the robot is manipulating.

- 'Learning' is building an abstract representation of sensor data. In the context of vision, it means to represent objects in such a way that they can be recognised and classified in a number of different scenes and poses.

- 'Classification' is the process of building a model of the objects perceived by the robot, labelling them into classes, and then identifying the correct class for new objects presented to the robot.

The main idea behind the hypothesis is that robots manipulate objects in their environment in order to better learn about them. Allowing a robot to make decisive actions on objects in its workspace means that it can learn to operate without human supervision. An example application is a robot which sorts objects in a factory. If the robot needs to work on a different set of objects, using current technology, a supervisor would need to provide data of the objects photographed in different environments. With a system which autonomously learns, the robot can move the objects itself and build visual models, reducing the amount of time needed in training.

The known motion of a robotic manipulator is used to localise and find the object as it is being manipulated. This allows the robot to focus only on the object that it is interested in, and remove data which only provide noise in the models, such as other objects or the robot itself.

The benefit of induced motion to the problem of *learning* is explored after the robot is able to segment objects. Making a plan to move an object allows the robot to obtain multiple views, in order to build a more robust model. The object can be isolated from other objects in the environment, removing the problem of other items occluding the robots view.

Finally, the combination of object segmentation and learning is explored, inheriting the benefits of both systems.

1.2 Motivation

An autonomous robot which can learn about objects is an interesting challenge. Despite the motivation to build machines that appear to operate like human beings, there are practical motivating factors, spanning from industry through to robotic maids to operate around the home.

Some examples and benefits of the application of autonomous learning to robotics are listed below:

- **Service Robots**: There is a large effort towards building service robots

Figure 1-1: Robots designed to operate with humans. From left to right, PR2 (Willow Garage), a personal service robot (Stanford), a teleoperated robot used to fetch a beer (Stanford), Dexter (UMass, Amherst)

that are able to operate around the home. The applications range from robotic care workers to help disabled patients to robotic maids for cooking. Figure 1-1 shows a selection of service robots with this purpose in mind.

The ability to recognise objects in the environment is essential for robots in this setting. The object models used for recognition could be learned from a set of images downloaded from the web or built from photographs taken by a human supervisor. Allowing the robot to manipulate objects and update its model autonomously makes it much more capable of operating by itself, particularly in environments where object models do not match the data well enough to make a conclusive decision.

- **Factory Robots**: Most of the robots in current use are found in industry. They serve a variety of different purposes, including sorting products, fitting parts, spraying, soldering and welding. All of these applications involve robot interaction with the environment and objects in the environment. The standard, and easiest, way to make the robot function in this way is to control the environment; pre-programming trajectories that the robot makes and delivering the objects to the correct place. A more desirable, and flexible system, would be a robot which can learn the product, or environmental object models, and have the robot move to the correct position autonomously. Showing that a robot can learn its own models makes training much more simple, an operator can give the robot the items it needs to learn about in order for it to build its own models.

- **Mobile Robots**: Intelligent machines can be used for mapping the environment, navigating through inhospitable areas or providing transport

Figure 1-2: Some example problems in recognising objects, taken from the CMU Ikea dataset [Hsiao et al., 2010]. The left image gives an example of an object which can be learned by robotic manipulation. The right three images show the object in and around the workplace, these images include variation in projective transformation, occlusion and lighting.

autonomously. Mobile robots can use a vision system to navigate. This requires some object recognition, the robot needs to understand when it has seen a particular landmark previously, in order to update its internal map. Although mobile robots, generally, do not have a manipulator to allow them to move objects, the robot is able to move its body in relation to landmarks in the environment. Navigation and mapping could then be enhanced, giving the robot the technology to choose trajectories which improve the learning process for visual landmarks.

- **Improved AI**: Beyond the practical applications of a robot which can learn about objects, are the implications for Artificial Intelligence (AI). The use of the body to improve a learning system is in support of the hypothesis that intelligence is embodied, that there is some level of simplification in having a body which can act on the world.

The examples discussed above touch on the need for robotics in the real world. Robots can be built to serve as operators in places that humans cannot go, for example, the disposal of nuclear waste, robots which operate in outer space, or under the sea.

1.3 Visual Learning and Robotics

One of the most fundamental problems in Computer Vision is to automatically separate objects into visual categories, building an abstract model for each category, and using the models to further categorise objects found in new

images. These methods can be applied to a scenario in which a robot needs to learn about objects in its environment, for example, a system for recognising and sorting objects moving along a conveyor belt.

The objective is to enable a robot to learn object models, in an attempt to be able to recognise them in new environments. Humans are able to perceive and recognise even when faced with a large amount of noise, occlusion, illumination and pose changes. These challenges make it very difficult for a machine to recognise objects. Human recognition has been studied in depth by psychologists for a long time. One of the founding researchers in the area, [Biederman, 1987], made the hypothesis that humans learn models by breaking objects into irreducible pieces or *geons*. Once the geons are found, their relative locations form a model of an object. Although the psychological concept of visual recognition has been studied in humans, building a replica on a machine is vastly complex; instead, models are built for vision systems which are to some extent, independent of the biological process of learning. Inspiration is taken from work such as [Biederman, 1987], as a principal foundation for visual learning and ported for use on a machine.

Since pure computer vision problems only deal with images that have been collected from a camera they usually don't assume any control over the environment. A robot has a means to interact with the world, plan and manipulate. Intuition implies that any extra information about the environment should be useful for the learning process. This thesis explores how manipulation and motion induced by a robotic manipulator benefits the process of perceiving objects and learning object models.

Despite human and machine capability to reconstruct 3D information from a pair of cameras, there are a set of problems which need to be accounted for when processing information to learn about objects. As objects move, their shape appears to change, due to perspective and scale. A change in lighting causes the object to become more saturated, and specular reflection can change the appearance of object parts significantly. Certain parts of an object may be occluded either by other objects in the scene, or because it is only possible to view the object from one angle, the object may also be partially out of view. It is desirable to design visual learning algorithms to be as invariant to these properties as possible. Figure 1-2 provides some examples of images which are

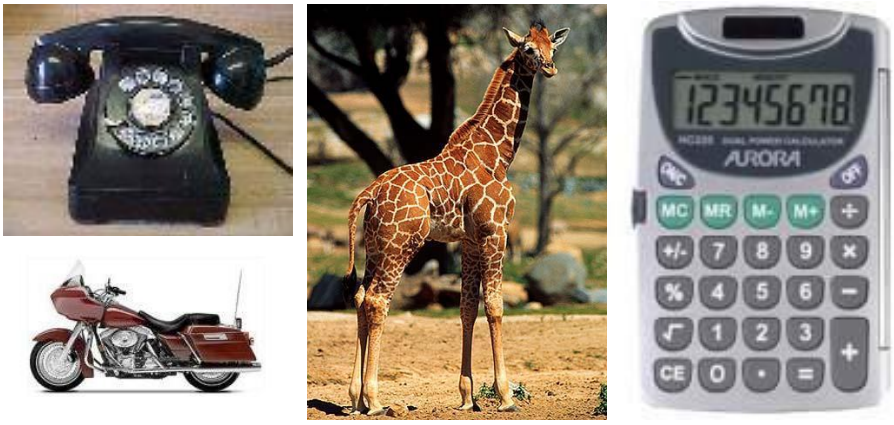


Figure 1-3: Sample images taken from the CalTech 256 dataset.

Figure 1-4: A set of images taken from a single class of the CalTech 256 dataset. The background, and pose of the giraffe, changes significantly between each image.

difficult to recognise because they vary in some of the properties mentioned. In principle, a robot can move an object and view it from all angles, acquiring all the information needed to build a model which is pose, illumination and occlusion invariant.

The affect of background noise when processing the visual data is important to consider. Unless the object can be properly distinguished from its background, before any learning takes place that background will have an influence on the object model. One way to minimise this affect is to take images with a range of different backgrounds. There will then be more variance in the background between the images than in the object. If the model is built

statistically, features which belong to the object will maintain a higher probability than background features, thus, *averaging out* the background. Figure 1-3 shows sample images from the standard computer vision dataset, Caltech 256. Figure 1-4 shows a set of images from a single class. It is clear that the background changes significantly with the object of interest changing pose between each image. In the case of a robot learning around the home, the background varies much less, unless the object is moved decisively. This advocates the use of manipulation to try to differentiate the background from the object.

To summarise the differences between current vision and robotic systems, exemplified by Figures 1-3 and 1-2, it can be seen that for current visual categorisation:

- The background changes significantly between each image.
- The object does not assume a single pose with respect to other objects in the environment.

These points are much different in a kitchen scene. The robot is able to observe the objects in their fixed position within the workspace and the background does not change much with viewpoint. Introducing a robot which can manipulate the objects and differentiate them from the background, while changing the pose, could greatly assist the robot in learning within these environments. The issues introduced above motivate the work explored throughout the thesis.

1.4 Current Robotic Vision Systems

The field of Computer Vision has produced a number of efficient and statistically robust algorithms for visual perception and learning using cameras. Most current robotic systems use the vision system independently of the robotic sensors or actuators. The addition of a robotic manipulator, provides a range of extra sensors, including position, velocity, acceleration and torque sensors. Where most current robotic systems do not use this information to aid visual learning, the results in this thesis pave the way towards a system which does.

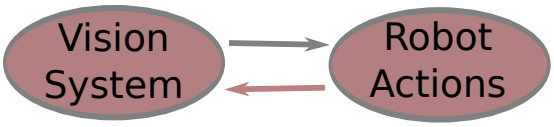


Figure 1-5: A graph showing the information flow in a robot. Models in the vision system can inform the robot of potential actions (black arrow), it is also possible for the robot actions to inform the vision system about how to perceive and learn (red arrow).

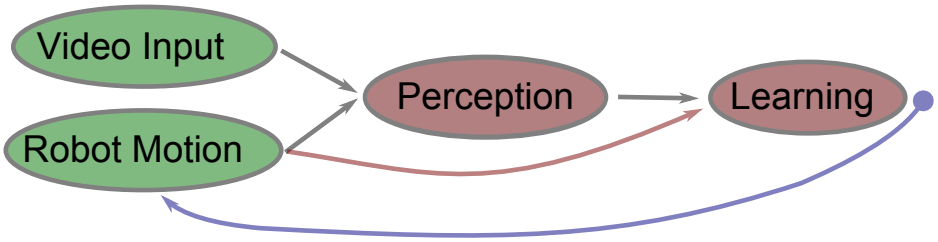


Figure 1-6: A diagram showing the process of learning with a robotic manipulator, the robot motion has information for the image processing algorithm and also the learning. Once object models are built, the robot can use them to make a plan about how to act on the objects further.

The decoupling between the vision and robotic sensors is inherent in most robotic systems. The robot observes an event and creates a plan to act based on what it has seen. Although the robot is constrained by the body it has, the physical properties and sensor readings have little to no impact on the decision made. Having the ability to purposefully change the environment gives a huge amount of information to the robots vision system. If the robot is holding an object of interest in its hand, most of the other visual information can be ignored. If it has moved something, knowing the motion of the hand gives information about how the object moved.

Figure 1-5 gives a graphical representation of the information flow in standard robotic systems. Vision software is used to give the robot information about how to move, in very few cases the robot movement is used to tell the robot how to 'see'.

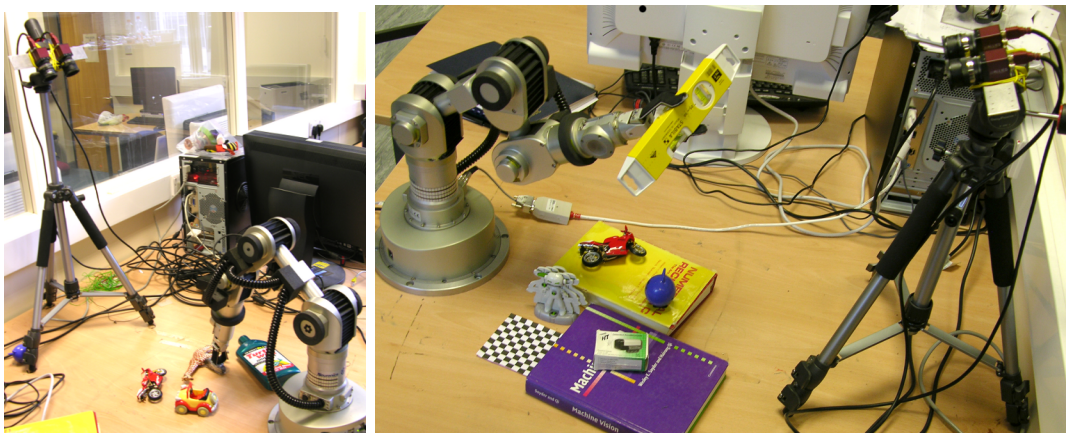


Figure 1-7: The experimental set-up. The image on the left shows the robot recursively pushing objects in the environment. The right image shows the robot holding an object and moving it in front of the camera system.

1.5 The Experimental Platform

The system proposed incorporates robotic *perception*, the low-level processing of visual information; and *learning*, the use of processed information to build a model of objects in the environment. In the experiments, the robot will be presented with a set of objects, which it will then decisively push. The vision information is processed to remove noise and background, while allowing the object to be viewed from multiple angles. Finally, a statistical object model will be built, allowing the robot to recognise the object in a new scene. To support the hypothesis, robotic motion and planning is used at every stage of the algorithm to improve or simplify the process. Figure 1-6 shows a graphical representation. The idea is that the robot perceives, learns and acts in a loop, building up a knowledge of the objects through interaction.

The robot is comprised of a 6 degree of freedom manipulator and a pair of Guppy cameras. The positions of the cameras are fixed relative to the robot. Changing their relative positions would require recalibration. A set of objects are placed in the workspace visible to the cameras and in reach of the robot arm. Figure 1-7 shows images of the set-up.

Although the experimental set-up is fixed, the algorithms outlined in the thesis are designed and tested to operate under various different conditions, including dynamic backgrounds. This allows the work to easily be ported onto a

more pervasive robot such as a humanoid operating around the home.

1.6 Contributions

The main contribution of the thesis is an autonomous and unsupervised system for learning and classifying objects using a robotic system. Although other methods are available for learning about objects, the system presented in this work contributes by providing extensions to current segmentation and learning algorithms. The result is a system which can iteratively plan, segment and recognise in order to classify objects with a high inter-class variance, while remaining robust to cluttered dynamic environments. The contributions of each chapter are summarised below, with references to the relevant papers.

- **A new visual-dynamic model for hand-eye coordination** [Beale et al., 2010]. The introduction of a new statistical model to incorporate dynamics and visual uncertainty for the problem of timed hand-eye coordination. The statistical model integrates mechanical information (forces) with tracking information (vision) to define suitable control trajectories. Trajectories can be designed to minimise energy consumption or maximise precision while intercepting a moving target. This work shows that robotic sensor data can be combined with vision to aid a common problem in robotics, the problem of hand-eye coordination. Although the work does not directly support the hypothesis, since it stands outside of visual learning, it provides a good case study in vision and robotics. It also gives all the technology needed to make the robot move in the environment, and exchange motion information between the robot manipulator and vision system.

- **Robust robot-object segmentation in the presence of dynamic and cluttered environments** [Beale et al., 2011a] contributes a new method to segment and localise objects based on robot's motion. Improvements are made to previous algorithms in the area when there is a moving (dynamic) background. The probabilistic formulation allows prior knowledge to be used when segmenting, something that has not been considered previously. This is in support of the hypothesis since it shows

that manipulator motion can be used to segment and localise the object that the robot is moving.

- **Online learning for robotics** [Beale et al., 2011b]. An online version of the successful batch topic model 'Latent Dirichlet Allocation' (LDA) is applied to classification in computer vision. A 'universal dictionary' is tested and applied, so that the algorithm is able to build topic models, without having to update the dictionary. The model is extended into a 'Mixture of Topics' (MoT) model, not previously used in computer vision since it requires object level segmentation to perform properly. The use of online LDA and its extension in to MoT for classification is also new to the robotics literature. The use of online learning does not contribute to the hypothesis, but provides a good platform for visual learning on a robotic system.

- **Self-supervision** [Beale et al., 2011b]. A new algorithm for unsupervised learning using a robot, based on a supervised Mixture of Topics classifier. The work supports the hypothesis by showing that inducing motion on objects allows the robot to supervise its own learning. Items from each class of object are labelled by the robot for use in a *supervised* machine learning classifier, removing some problems associated with unsupervised visual learning. The robot can recursively manipulate objects in the workspace, exposing multiple views, and reducing noise caused by specular reflection.

The work concludes in a system which combines segmentation and classification to recursively learn about objects, inheriting the contributions of each system. This is presented in the final chapter of the thesis, and provides a novel method for learning about objects on a robotic system. The system is robust to cluttered and dynamic environments, while allowing the robot to view objects from a number of poses. The results show the system to robustly categorise objects placed in the workspace.

1.7 Structure of the Thesis

- **Part I - Introduction**
  - **Chapter 1: Introduction**
    This chapter provides an introduction to robotics and the problem of machine learning. The thesis is motivated by the problem of building a robot which can effectively learn about objects in the environment, under varying and uncertain conditions.
  - **Chapter 2: Background**
    In this chapter the history of object learning in robotics is explored. The fundamental problems of computer vision, and how they have been solved in general; the use of computer vision in robotics ; most relevant to this thesis, how the literature uses robotic manipulation to aid visual perception and learning.
  - **Chapter 3: Vision and Manipulation** *Vision, motion and dynamics.*
    This chapter shows how the robot and vision system are calibrated to each other. A case study is presented which explores a combined model of the uncertainty in the vision system and a dynamic model of the robotic arm. This gives the basis for a robot to move objects in the environment, while providing the first step towards a model which uses both vision and robot sensor readings to operate efficiently.
- **Part II - Robot Perception**
  - **Chapter 4: Object Segmentation** *Manipulation aids perception.*
    In this chapter the robot pushes objects in order to localise and segment them from their background. This provides a simple approach to figure / background segmentation, and improves current methods, allowing the robot to segment when the background is changing.
  - **Chapter 5: Experiments in Object Segmentation**

The algorithm is experimentally compared to previous work in the area, showing an improvement in dynamic / moving environments. There are also results showing the algorithm working for three dimensional robotic motion.

- **Part III - Learning**
  - **Chapter 6: Autonomous Visual Learning of Object Models** *How robots learn object models.*
    A major result in learning for computer vision is the bag-of-words model. This chapter explores the use of Latent Dirichlet Allocation (LDA) in a robotic scenario, allowing object models to be built Online, without the need to store or recompute old data. The results experiment with the idea of a wide, or universal, dictionary; built using a well known dataset with a large number of images. The chapter concludes with an unsupervised algorithm capable of learning object models on a robotic vision system.
  - **Chapter 7: Mixtures of Topics and Self-Supervision** *Manipulation aids learning.*
    The LDA algorithm introduced in the previous chapter is extended by the idea that objects are composed of multiple views, rather than just one. A method of self-supervision is explored for the robot, so that it can take control of its own learning, viewing objects from multiple angles and autonomously labelling input data. The algorithm is tested with data taken from the robot, and also on a well known publicly available dataset.
  - **Chapter 8: Combined Segmentation and Learning** *Perception and learning combined.*
    In this chapter object segmentation is combined with model learning. The results show an improved rate of classification in general, but the algorithm also inherits robustness to changing and cluttered backgrounds.
- **Part IV - Conclusions**
  - **Chapter 9: Conclusions**

Conclusions are drawn about the learning methods with a discussion about future work.

1.8 Table of Notation

This table provides the common notation used throughout the rest of the thesis.

Q	$\mathbb{R}^n \to \mathbb{R}^n$ Motion transformation function
Φ	Probabilistic motion model
ϱ	$\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ Error measurement function
$p(\varepsilon \mid Q, \Omega)$	$\mathbb{R} \to [0, 1]$ Error probability density function
Ω	Prior data relating to error distribution
ε	Error value
$\mathcal{S}$	Set of segments in an image
S_j	the jth segment in an image
P	$\mathbb{RP}^3 \to \mathbb{RP}^2$ Camera calibration matrix
I	Identity matrix
Σ	Covariance matrix
$N()$	Normal distribution
$Dir()$	Dirichlet Distribution
$Multinomial()$	Multinomial Distribution
α	Parameter controlling the distribution of documents
β	Matrix of multinomial parameters controlling the distributions of words given a topic
θ	A document, or proportion of topics
z	A single topic index
w	A word index

Chapter 2

Background

This chapter presents an analysis of the literature concerned with visual learning on a robotic system, in particular, how a robot can learn the visual appearance of objects in its environment. The problem of visual learning is large and detailed, covering topics throughout psychology, biology, computer science and mathematics. The background presented here is concerned with learning using a vision system and a robotic manipulator, providing the motivation for the contributions made in this thesis towards machine perception, learning and classification.

The literature is reviewed by finding robotic systems which are capable of learning, and building object models. The problem is then broken down into the components that are needed in order to build such a system; segmentation and learning. The emphasis is towards robots which use other sensor data, such as encoder readings, in combination with the visual input.

- **Segmentation**. The problem of breaking images into regions of pixels is used in robotics as a first step towards identifying objects in images. The literature surrounding object segmentation is reviewed, specifically to do with algorithms which only use pixel information and robotic sensor data.
- **Learning**. The information from the vision system can then be used to build a model of objects in the environment. The emphasis is on robots which can learn without the need for human supervision, and also, are able to update their internal models without the need to recompute old data.

Both of these problems are largely studied in the computer vision and

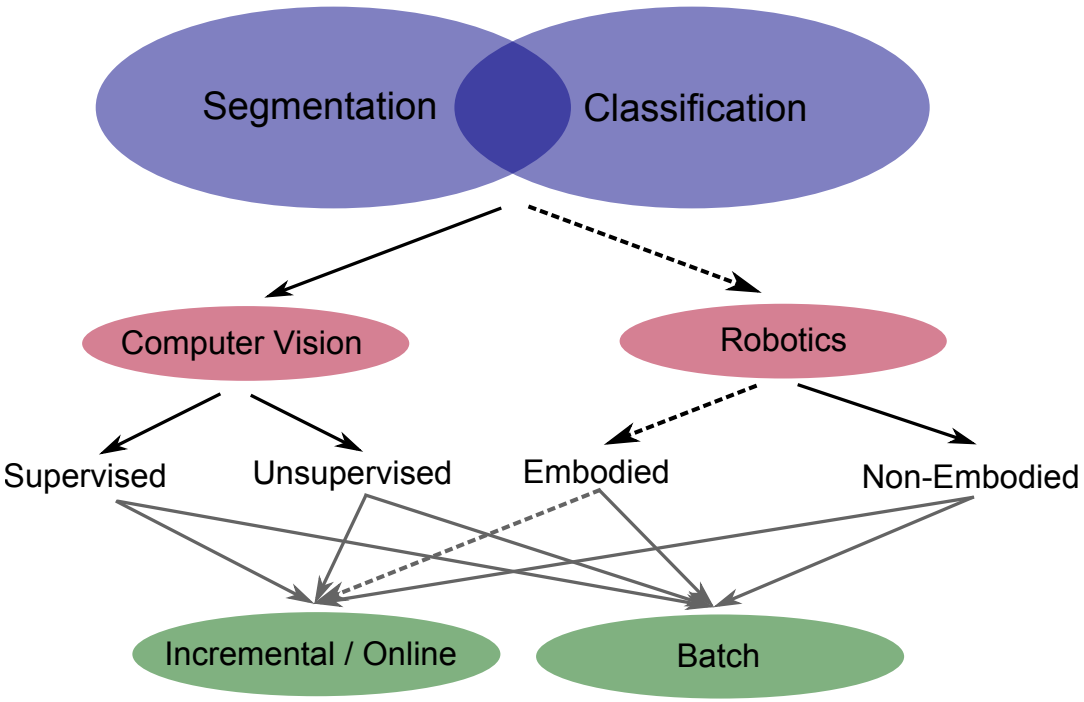


Figure 2-1: Diagram showing the structure of the literature. On the top level segmentation and classification are distinct, although some vision techniques involve solving segmentation and classification simultaneously. The literature is broadly separated into computer vision and robotics, where robotic systems have the ability to act on the environment. The work in this thesis follows the dashed arrows, and lies in between segmentation and classification.

robotic literature already, in both a general setting and also for more specific problems. The background covered in this chapter draws attention to literature in which:

1. **Robotic motion**, or knowledge of the robot position is used to segment and localise objects.

2. The learning is **online**, with the robot updating its model as new data becomes available.

3. Robots can learn visual object models autonomously, with little or no human supervision.

Figure 2-1 is a diagram showing how the literature is structured. Both segmentation and classification are large and unsolved problems in vision, and

applicable to the fields of robotics and general computer vision. The algorithms can solve the problems in a *supervised* manner, taking input from the human user to aid the results; or *unsupervised*, without any external input. In the robotics literature, work is split by its use of extra sensor information to aid the vision. Finally, methods are differentiated into *incremental* or *online*, and *batch*. Batch methods require all of the data in order to compute classes or segments, whereas online methods can update the models without storing old data.

The chapter begins with general learning on a robotic system, and moves to more specific areas of robotic perception and learning. The literature is related to the hypothesis stated in the introduction and also the contributions of the thesis.

2.1 Learning in Robotics

Allowing a robotic system to learn is generally desirable, as it allows the robot to easily adapt to new environments. In the context of machines, an algorithm which changes its behavior based on its observation of the input variables, is known as a *learning* algorithm, which forms a part of the larger set of *non-deterministic* algorithms.

The review in this section gives an overview of machine learning, and its application to a few of the many problems in robotics. The general problem is covered along with a few more specific examples. This gives a motivation and basis for the approaches used throughout the rest of the thesis.

- **General Machine Learning**. *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"* – [Mitchell, 1997]. Machine learning is the problem of building computer programs which can adapt their response to an input, according to a given set of data, or measure of performance. Since sensor readings are noisy, and the world is complicated and unpredictable, most machine learning algorithms are based on statistics and practical probability theory [Bishop, 2006]. The applications range from computer vision, through to bioinformatics and search engines.

- **Learning to Navigate** A large amount of research goes into creating robots which are able to navigate the environment. The general area is known as Simultaneous Localisation and Mapping (SLAM) [Davidson, 1998]. Robots equipped with SLAM observe the environment using vision or a selection of other sensors, and build a map, while localising themselves within the map. This kind of navigation can require machine learning, so the robot is able to identify landmarks that it has seen before [Gibbens et al., 2000, Kaess et al., 2011, Strasdat et al., 2010]. The techniques employed from machine learning can be used to navigate a car along a road, [Thrun et al., 2006]; allow a submarine to navigate underwater [Ribas et al., 2010], in an environment in which GPS does not work; or for aerial vehicles [Conte and Doherty, 2008].

- **Learning to Control**. Alongside the ability to navigate, robots also need to control their motors, wheels, propellers, or joints. For instance a humanoid robot may need to learn the motor trajectories required to pick up an object [Schaal et al., 2003]. The dynamics of the robotic manipulator can also be learned, including masses of each link and coefficients of friction etc. This allows more flexible software to be written for a range of different robotic platforms [Petkos, 2008].

- **Learning about Objects**. This thesis focuses on a robot which can autonomously learn about objects in the environment. The work draws on techniques from machine learning, motor control, text analysis and other fields, in order to obtain results. Object classification and recognition span many fields and applications on their own, for instance they can be used in SLAM to identify landmarks [Gibbens et al., 2000], on search engines to find similar images [Ma and Manjunath, 1997], and also for general understanding of scenes [Alex Flint, 2011].

- **Learning from Human Psychology**. Machine learning also addresses the problem of general intelligence. Given sensory information and a means to act upon the world, the robot needs to decide how to make decisions. The area of developmental psychology sheds some light on the way humans develop through childhood [Wood et al., 2005]. These ideas have been explored using a robotic system [Vernon et al., 2007], with an

emphasis towards learning about objects [Fitzpatrick et al., 2008]. The general goal of this area is to enable a machine to progress through one stage of development, in the same way a human would [Asada et al., 2009].

Throughout the rest of the chapter the literature is reviewed specifically in the context of object classification using a robot and vision system. The thrust is to find general algorithms in computer vision which are able to build object models for classification and recognition, and also, to find a means to build such a system from the robotic literature. The hypothesis relates the research to embodiment, or the knowledge of the robots movement to aid the perceptual process, and the use of object segmentation and localisation in learning.

2.2 Visual Perception and Learning

The first goal is to understand the vision system, and how to build a computer system which is able to learn about objects and classify them. There are several ways to attempt a solution to this problem, but the approach taken throughout the thesis is to break it into two subsystems; segmentation (visual perception) and learning. Literature is reviewed in both of these areas throughout Computer Vision and Robotics.

2.2.1 Segmentation

Segmentation is a fundamental problem in computer vision. The general definition is to break an image or video into semantically disjoint regions. This definition is useful but relatively ill posed as, in order to segment, a good definition of 'semantic' is needed. In a top-down segmentation algorithm, prior knowledge about the objects in the scene is assumed, the image is then broken up according to known object models. In bottom-up segmentation, no prior object knowledge is assumed, the algorithm breaks the image up according to the data, coherent regions of colour, motion or texture.

The computer vision literature broadly classifies the segmentation literature into two areas,

- **Bottom-up** Segmentation is the use of raw image data and simple local

		Segmentation	Classification	
Vision	Image	[Arbeláez et al., 2009] [Martin et al., 2004] [Schaeffer, 2007] [Bilmes, 1998] [Felzenszwalb and Huttenlocher, 2004] [Paris and Durand, 2007] [MacQueen et al., 1967] [Boykov and Kolmogorov, 2004] [Comaniciu and Meer, 2002]	Online	[Banerjee and Basu, 2007] [Canini et al., 2009] [Wu et al., 2008] [Fu et al., 2010] [X. Song et al., 2005] [Li and Fei-Fei, 2010] [Nistér and Stewénius, 2006] [Griffiths and Steyvers, 2004]
	Video	[Huang et al., 2009] [Stein et al., 2007] [Torr and Zisserman, 1998] [Paris, 2008] [Horn and Schunck, 1981] [Torr, 1998] [Brox and Malik, 2010] [Ross, 2000] [Weiss and Adelson, 1996] [Vazquez-Reina et al., 2010]	Batch	[Weber et al., 2000] [Fei-Fei and Perona, 2005a] [Sivic and Russell, 2005] [Griffin and Perona, 2008] [Philbin et al., 2010] [Yuan et al., 2007] [Zhao et al., 2010] [Su et al., 2009] [Thomas et al., 2006] [Lazebnik et al., 2006] [Sudderth et al., 2005] [Agarwal and Roth, 2002] [Amit and Geman, 1999] [Fergus et al., 2010] [Winn et al., 2005] [Fergus et al., 2003] [Tuytelaars et al., 2010]
Robotics	Robotic Motion	[Arsenio et al., 2003] [Fitzpatrick et al., 2008] [Metta and Fitzpatrick, 2003] [Kenney et al., 2010] [Beale et al., 2011a]	With Robotic Segmentation	[Welke et al., 2010]
	Kinematic Model	[Mishra et al., 2009] [Mishra, 2010] [Ramanathan et al., 2010]	Plain Background	[Beale et al., 2011b] [Borotschnig et al., 1999] [Paletta and Pinz, 2000] [Chen, 2008] [Croon and Postma, 2006] [Williamson, 2009] [Willimon et al., 2010]
	No Robot Motion	[Katz and Brock, 2011] [Katz and Brock, 2008]		

Table 2.1: A table showing how the literature is separated. Distinctions are made between vision and robotics; and also, segmentation and classification.

Figure 2-2: Different static low-level segmentation algorithms. Left: The original image; Centre: Mean Shift; Right: Berkely

image metrics to group regions of an image; for example, grouping pixels which are close together and have a similar colour.

- **Top-down** Segmentation is used when there is some prior knowledge of the objects in the environment, an object model. The models are used to identify the objects.

In this work, the primary focus is on bottom-up segmentation, since it is widely used for robot perception. The assumption is that the robot has little or no understanding of any objects in the environment. Literature is also broken into video and image segmentation, the similarity metrics used for image segmentation are based on colour or texture information; whereas video segmentation can use motion. It is certainly not the case that objects are similar in colour or texture, but the assumption that objects move coherently is more accurate. Bottom-up video segmentation forms a core part of robot perception and is the focus for review. The categorisation is reflected in Table 2.1.

The most basic approach to bottom-up image segmentation is clustering. Each pixel in an image is composed of red, green and blue intensity values. Assuming we want each region of the image to contain coherent colour information, the pixels are clustered according to their position and colour

intensity. For a single image, these values are clustered using an algorithm such as mean shift [Comaniciu and Meer, 2002], k-means [MacQueen et al., 1967] or graph partitioning [Schaeffer, 2007]. If an extra assumption is made about the distribution of the data in each segment, then the results can be improved. In the case of Multivariate Gaussian distributed clusters the EM algorithm [Bilmes, 1998] can be used to refine the clustering. This is not necessarily the case in a general image, and so discriminative clustering works well.

The main concern of clustered approaches to image segmentation is the amount of time they take to compute. Algorithms exist to reduce the overall complexity, the most notable being [Paris and Durand, 2007] who use a topological method to improve speed. Building a graph over the image pixels and reformulating segmentation as a max-flow min-cut problem [Boykov and Kolmogorov, 2004] can also be used to efficiently segment an image into colour or texture coherent regions [Felzenszwalb and Huttenlocher, 2004].

One of the most recent and popular methods for segmentation is from Berkely [Arbeláez et al., 2009, Martin et al., 2004]. They base the technique on the edges present in the image. A model is trained from a set of ground truth images in which human participants are asked to break the image into the regions they believe partition the image. An edge detector is trained on the ground truth images, which is used as a base model for segmenting unseen images. However, this algorithm is arguably top-down, since some prior knowledge is present from the human participants.

Figure 2-2 shows standard Mean Shift in comparison to the Berkely segmentation algorithm. The Berkely method is much stronger at accurately finding the edges, and all segments are strictly subsets of objects, making it a suitable algorithm for image analysis. The trade off is the amount of time it takes to compute the segmentation, for which the Berkely method's performance is much worse.

The techniques described so far only take single images as input. When dealing with video, each frame can be independently segmented, but the boundaries of each segment are not continuous between frames. The solution to this problem is to consider multiple frames of a video when segmenting. The methods for static image segmentation can be extended in order to segment videos by considering them as a volume of colour intensity. In this case each

pixel is represented instead by a 6 dimensional vector, an extension of segmentation for a single image, including time as an extra variable. This is known in the literature as volumetric segmentation [Grundmann et al., 2010, Apostoloff and Fitzgibbon, 2006]. Mean-shift can also be used for volumetric segmentation, but is generally quite slow, the method of [Paris, 2008] makes the process faster using Morse Theory.

A good way to represent motion in a video is to use optical flow [Horn and Schunck, 1981]. Optical flow represents the motion throughout the video. Without considering the colour intensity of each pixel, optical flow can be used to segment objects assuming they move according to some motion transformation (affine, translation, projective) [Weiss and Adelson, 1996]. If there are multiple objects in the environment which move according to different distinct transformation functions then probabilistic modelling can be used to segment the objects and also estimate their motion [Torr and Zisserman, 1998, Torr, 1998]. Using a method to track features throughout a video, whole trajectories can be clustered to segment videos in to regions of coherent motion [Brox and Malik, 2010, Vazquez-Reina et al., 2010].

Motion segmentation algorithms are useful on a robotic platform if they are fast to compute or, ideally, incremental. The video is broken into regions that are likely to contain objects, which can then be used for higher level learning. If the robot is manipulating an object, or has some control over the environment, the extra information from the robotic sensors can be used to segment *and* localise it, or in other words, extract the object of interest from all other objects in the environment. In the following section literature is reviewed from the robotics community, who attempt to solve this problem.

2.2.2 Interactive Robot Segmentation

The literature discussed so far is for general bottom-up segmentation, where there are few assumptions made about the objects in the environment. When using a robot, the system does not only have extra information from the range of sensors on its body, but also has control over the environment. The work presented here focuses on systems which use extra sensor information to aid the segmentation process. Algorithms which use sensor information to segment

objects have an advantage over ones which don't, since they are also able to *localise* the object that the robot is moving. This means that the robot can differentiate the object of interest from other objects in the environment, and also the robot body.

Bottom-up segmentation in robotics can be separated into the following categories, depending on the information used by the robot:

- **Proprioceptive Information**. It is well known in the literature on human development that young children develop a sense of *proprioception*, awareness of the relative body positions, before beginning to learn about objects in the environment. Along with basic image processing tools, this gives a child the tools it needs to begin learning about objects. When applied to robotics, the word means that the system has a prior model of its own body (possibly a CAD model) which can be used to remove the body from the images.

- **Robotic Motion**. Another way to segment objects is to use the motion that the robot makes. The differentiation between using proprioceptive information and robotic motion is analogous to the vision problem of image and video segmentation. In the former case, the robot uses information for a single instance in time, in the latter, knowledge of the robot motion from more than one frame is used in the segmentation.

- **No Sensor Information**. In the final category, no robotic information is used at all. Methods in this area either use a prior object model of some kind, or assume that the robot is operating on a plain background.

This categorisation is reflected in Table 2.1, showing the different papers and their use of the robot's sensors.

One way to use the robot to aid segmentation is to move the cameras to focus on a position inside the boundary of an object [Mishra et al., 2009, Ramanathan et al., 2010, Mishra, 2010]. This work requires some weak prior knowledge of the object's location before segmenting. Since the segmentation is based on edge maps, complicated objects with strong edges and patterns on it may be segmented incorrectly. The work is classified as proprioceptive since the robot can segment with only one frame of the video. Figure 2-3 gives an example of this kind of segmentation.

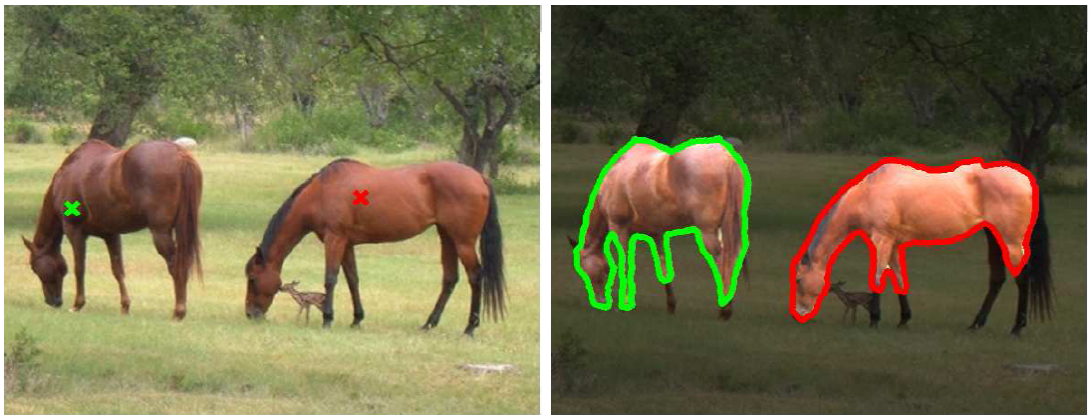
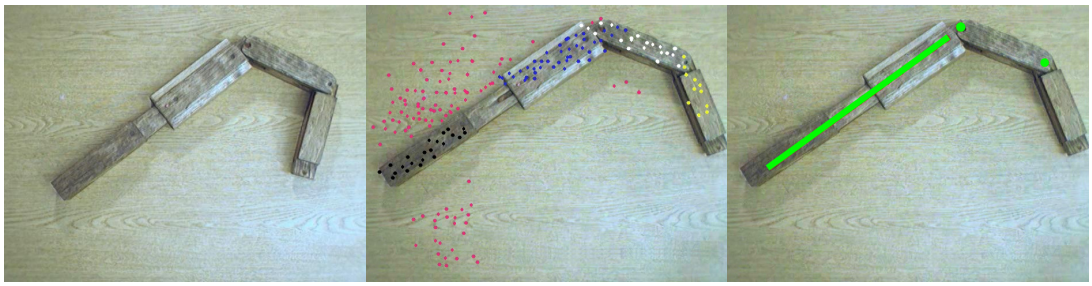


Figure 2-3: Segmentation by fixation. The robot fixates on a point in the scene and segments the object in that region. Images taken from [Mishra et al., 2009].

The first papers written to use robotic motion to aid segmentation were written at Massachusetts Institute of Technology (MIT) in the artificial intelligence laboratory [Fitzpatrick and Metta, 2003, Fitzpatrick, 2002, Metta and Fitzpatrick, 2003]. The robot COG pushes objects and uses frame differencing along with knowledge of the frame in which the object first moves to segment it from its background. This process involves no learning or the need for any prior knowledge, the assumption is that the robot is able to touch (prod) an object. The segmentation is computed over three frames. In the first pair of frames only the robot is moving, and in the second pair of frames the robot has pushed the object. The algorithm uses a combination of image subtraction and graph cuts [Boykov et al., 2001] to both remove the arm and determine the object from the background. The use of image subtraction means that any objects that are moving other than the arm and object causes the segmentation to fail.

Assuming that the object is being held by the robot, optical flow can be used instead of frame differencing [Arsenio et al., 2003]. This requires a kinematic model of the robot arm to remove it from the segmentation, and also requires that the object is held by the robot to begin with.

More recently, the idea of using interaction to improve or simplify object segmentation has been extended [Kenney et al., 2010]. This work focuses on the ability to not only segment the object but build a model, and track it



(a) Video Frame (b) Segmented Points (c) Kinematic Constraints

Figure 2-4: Segmenting object which have articulated motion, the kinematic constraints are calculated from the resulting segments [Katz and Brock, 2008]

throughout successive frames of the video. The underlying segmentation algorithm is based on the subtractive method by [Fitzpatrick and Metta, 2003], and is prone to the same shortcomings.

If sparse (trackable) feature points are used instead, then the graph cut algorithm first used by Fitzpatrick can be extended for articulated motion [Katz and Brock, 2008]. Figure 2-4 shows qualitative results from the algorithm. Feature points are extracted from frames of the video, and if the motion of the object is articulated, the algorithm will calculate the kinematic constraints between each link. The points are then grouped into segments which conform to each constraint. This work can also be used in 3D, for service environments such as in the home or workplace [Katz and Brock, 2011]. The algorithm works by saving an image of a scene without the robot, manipulating the environment, and then saving another image. Although the robot has had some interaction with the environment, the movement is not used in the segmentation. Thus, this work is classified as segmentation without the use of robotic manipulation.

2.2.3 Object Classification

Object classification is the problem of identifying a visual category for a given object automatically. The categories are pre-learned from a set of training data, according to a particular statistical model. In computer vision the classification algorithm is given a set of training images, each containing a number of objects. The algorithm should reduce the amount of information in the image to a level general enough to classify images in the presence of noise and occlusion, with

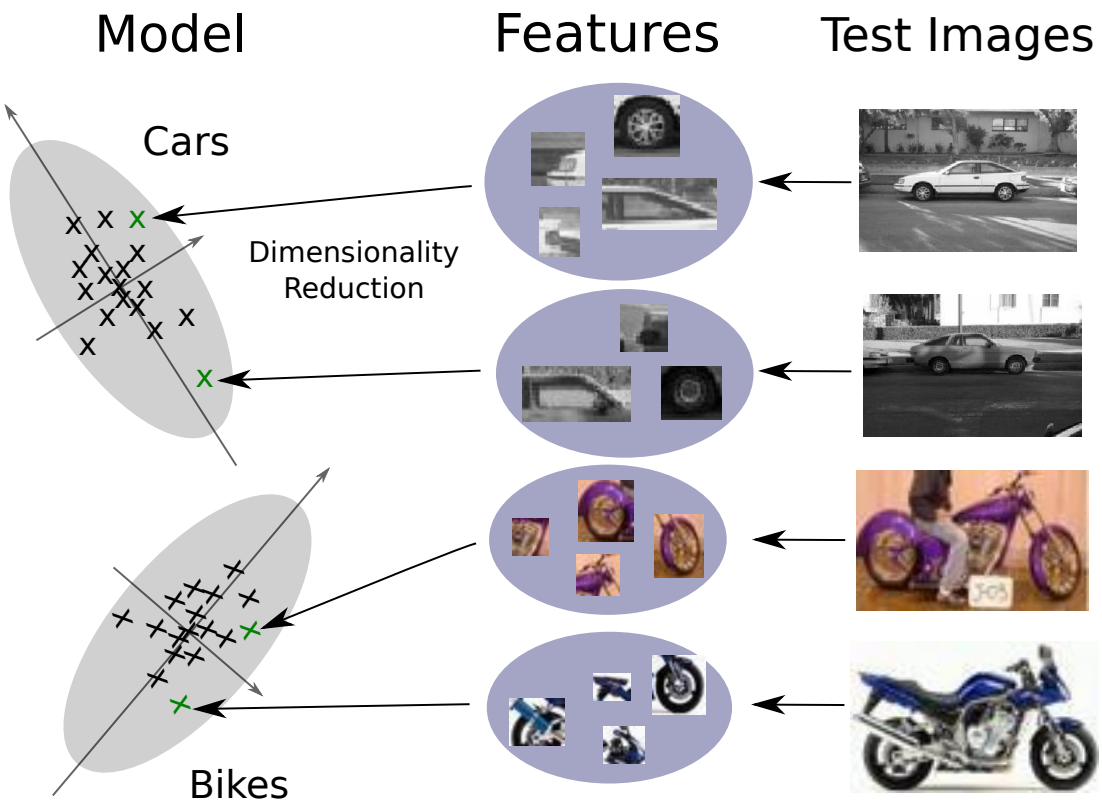


Figure 2-5: A general classification algorithm, images (right) are turned in to features (centre). Some form of dimensionality reduction is used on the set of features to turn them into vectors. A statistical model is built over the vectors, allowing new images to be classified.

invariance to transformation. A number of these problems are reduced by choosing a good set of features to represent the images.

Once a set of features has been chosen, a statistical model is built to represent the content of the images, with the purpose of reducing dimensionality in a way that separates the numerical representations of objects between classes. Figure 2-5 represents an ideal classifier, features in the image are reduced in to 2 dimensional vectors. Images in the classes 'car' and 'bike' are well separated, so that a Gaussian model can be built to represent the data, and classify new images in the respective class.

Learning algorithms are discriminated into *batch*, *online* and *incremental* approaches, explained in more detail below:

- **Batch** approaches to learning which compute categories over a fixed

training set. If new data is added to the training set, then the categories are recomputed over both old and new data.

- **Incremental** approaches, where only the new data needs to be computed but the whole training set must be stored.
- Alternatively, **online** approaches only require the new data to be stored and computed while the model is updated.

Computer Vision has a lot to offer to the problem of visual classification, covering a vast range of different methods and approaches. Recent work has seen a diverse range of developments including learning taxonomies of classes [Griffin and Perona, 2008, Winn et al., 2005], learning spatial patterns of words [Philbin et al., 2010, Yuan et al., 2007, Zhao et al., 2010], learning tens of thousands of categories [Fergus et al., 2010], and learning 3D objects [Ferrari et al., 2010, Su et al., 2009, Thomas et al., 2006]. The literature search throughout the rest of this chapter is restricted to Bag of Words (BoW) approaches, since it is the general approach used in the thesis. The main focus is on online learning, and is discussed in the following section.

Bag of words

The 'bag of words' (BoW) model dates back to the 60's [Maron and Kuhns, 1960] for analysing text documents. The model is based on the simplifying assumption that words in a document are unordered, and that a document can be represented by the frequency of the words within the document alone. Mathematically speaking, we assume that each word is statistically independent of other words in the document.

This idea relates to the problem of object classification: If an object can be represented by a set of low level features (words), then a bag of words model represents the object as the frequency of features contained within it, e.g. a car has two wheels and four doors, where as a bike has two wheels and zeros doors. As a result, an image is represented as a histogram over a fixed word base, and an example is shown in Figure 2-6. A detailed explanation of BoW can be found in Appendix B.1.

Bag of words approaches have begun to take use recently in computer vision [Agarwal and Roth, 2002, Amit and Geman, 1999, Burl et al., 1998, Fergus et al.,

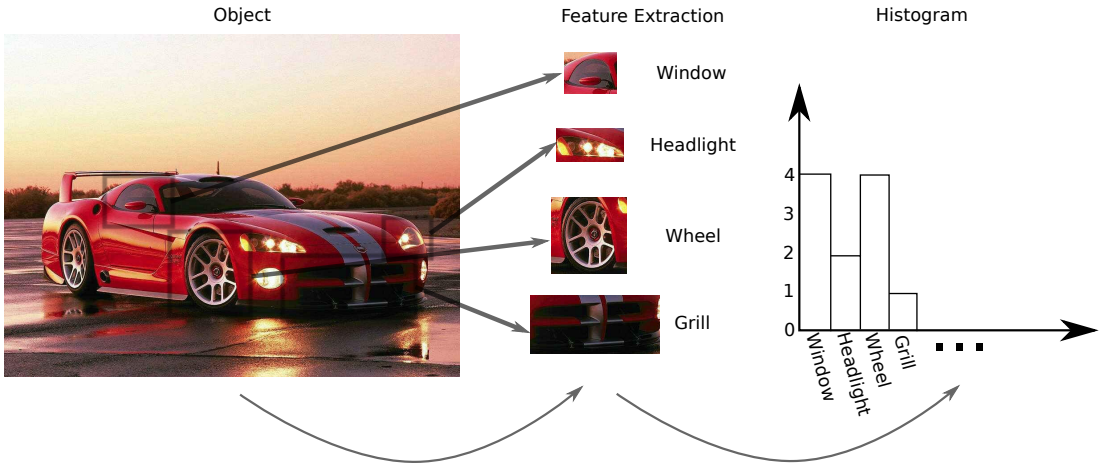


Figure 2-6: A diagram showing the steps involved in a bag-of-words model. The far left is an image of a car, features are extracted from the image to form 'words'. The image is finally represented as a histogram of the features in the image over the word base.

2003, Weber et al., 2000]. The use of robust image features [Lowe, 2004a] improved classification considerably, and has lead to more modern algorithms being developed, with improved results in [Fei-Fei and Perona, 2005a, Sivic and Russell, 2005]. This work provides a basis for the results found throughout the thesis, since features are quick to compute using a GPU, making a practical robotic implementation more feasible.

In problems, such as a robot in a free environment, it is important to learn from images that show more than one object at a time. The use of latent variable models shows best results for this [Tuytelaars et al., 2010]. For example, [Sivic and Russell, 2005] use probabilistic Latent Semantic Analysis (pLSA) to learn object classes, Fei-Fei and Perona use Latent Dirichlet Allocation (LDA) [Fei-Fei and Perona, 2005a]. [Lazebnik et al., 2006] use a pyramid scheme to learn about whole scenes, and [Sudderth et al., 2005] use a hierarchy of Dirichlet Processes.

All of these methods are known as *Batch* learning algorithms. The training set with a fixed size is provided. If more training data becomes available to the robot, it can't update the object models without reprocessing all of the previous data. The next section reviews methods which are able to update their models when new data and objects become available.

Online and Incremental Learning

For a robot to learn in an ubiquitous environment, with several new objects entering and leaving the scene, it is important that the learning algorithm has the capacity to update its model as new data becomes available. A distinction is made between *incremental* learning, where the algorithm can keep all of the data used throughout the training, and *online* learning, where old data can be thrown away. Online learning is more desirable on a robotic platform as the storage requirements and processing power are typically limited.

When applied to bag-of-words there are a number of other problems that need to be considered in order to make the system work online. Although the statistical model can be incremental or online over the histograms, the number of words may change. The literature deals with this problem in a number of ways, either by building a dictionary which is as universal as possible, or by changing the dictionary online while taking care to update the BoW model.

Probabilic Latent Semantic Analysis was developed before LDA and has been extended to online learning, with the application of automatically selecting text questions [Wu et al., 2008]. The algorithm has also been used for recognising visual objects [Fu et al., 2010]. This work allows the vocabulary and classes to change at the same time, with the vocabulary length changing as data becomes available.

Online LDA has also been used in computer vision [Li and Fei-Fei, 2010], with the use of a Hierarchical Dirichlet Process (HDP) for automated online picture collection. The work is motivated by the need to search the web for new images, and the results are shown to compare well with batch learning. There are examples of online LDA studied throughout the machine learning literature [Griffiths and Steyvers, 2004, Banerjee and Basu, 2007, Canini et al., 2009, X. Song et al., 2005], but they are not directly applicable to computer vision.

The algorithm chosen for use in this thesis is a direct implementation of [Hoffman et al., 2010a], but adapted for use in computer vision. The vocabulary is of a fixed size, and computed before the LDA model. The details are covered in Part III.

The approaches mentioned above consider the problem of general machine learning, applied to computer vision problems. For a robot operating in a

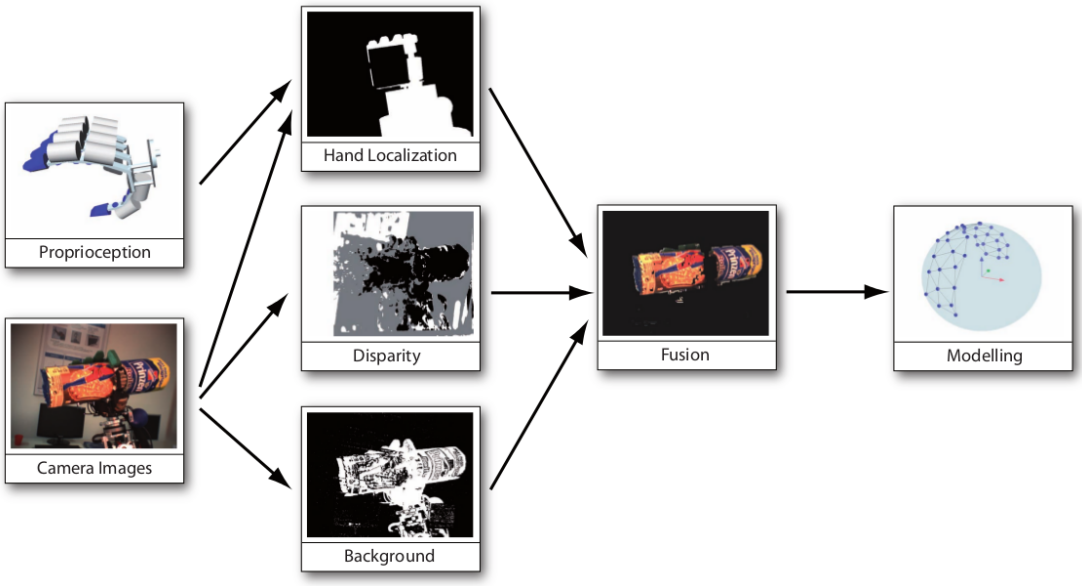


Figure 2-7: The learning system developed in [Welke et al., 2010]. The robot grasps an object and segments it from the background. After segmentation, a statistical model is built that allows the robot to identify the object in a new scene.

workspace, there are not only a different set of challenges to face, but also more sensors and control over the environment. For example, a robot working in a kitchen environment does not have access to images containing objects on different backgrounds, rather a single scene which does not change unless manipulated. The robot is able to move around the environment, change it, and also read from motion sensor placed around its body. The following section covers visual learning in robotic systems, and approaches which use a combination of vision, manipulation, and robot motion for classification and recognition.

2.2.4 Active Robotic Learning

The previous section covered algorithms which can learn from a general set of images. Using a robot to learn about object categories introduces a different set of problems. In a general setting, such as an industrial workplace, objects do not move by themselves. This means that, in order for a robot to see objects from multiple views, it must change its position relative to the object. The

most straightforward way to achieve this is to use the robotic arm to push the objects. However, the problem remains of distinguishing the object being manipulated from others in the environment.

Literature in this area is categorised into methods which;

- **Use robotic segmentation** as a basis for discriminating objects from their background.
- **Assume a plain background**, or **use prior knowledge** to detect the objects which are being learned about.

The idea of orienting a camera system relative to an object of interest, in order to learn about it, dates back to the 90's [Borotschnig et al., 1999]. In this area, objects are placed on a platform which rotates, the angle the object makes with the camera is used to build a robust probabilistic model. This is known as *active vision* [Paletta and Pinz, 2000, Chen, 2008, Croon and Postma, 2006], and is useful for a robot which can hold an object in a fixed known position relative to the camera system.

The area of active vision makes the assumption that the object can be well segmented, usually assuming a plain background. Although this doesn't imply the necessity of segmentation, algorithms which learn based on these methods require a decent segmentation in order to work properly. This suggests a need for object segmentation in the robotic learning process.

An object's shape can be used as a visual description; and a good object segmentation provides one of the most basic object models. Object segmentation for robotic platforms was discussed in more detail in Section 2.2.2, covering state of the art robotic segmentation algorithms [Fitzpatrick and Metta, 2003, Fitzpatrick, 2002, Metta and Fitzpatrick, 2003, Arsenio et al., 2003, Kenney et al., 2010]. Although these methods can be used in a system which learns about objects, they do not cover the process needed to build an object model, or verify the models in a classification or recognition task.

Object classification has been studied for robots which are able to manipulate or grasp objects in the environment [Li and Kleeman, 2011, Li and Kleeman, 2009]. This approach uses visual cues to infer the position and shape of the object in order to pick it up. Once the object has been grasped, the features on the object are recorded and used to match the object again in novel

images. The initial segmentation is based on the symmetry of the object. Although the approach is shown to work well in a recognition task, it is limited to the small class of symmetric objects.

Making the assumption that the robot is already holding the object requires a different set of tools to segment and learn. Once the object is held, it is possible to move it along a path so that all sides become visible. Methods to generate trajectories, with the application of learning have been studied by [Ude et al., 2008, Omrcen et al., 2007], moving the object to the centre of the visual field and rotating it about its centre of mass. This work is extended by [Welke et al., 2010], who use a mixture of cues to segment the object from the background, and building the object model. Figure 2-7 shows the process used by [Welke et al., 2010] to learn the models, combining proprioceptive segmentation with learning.

More recently, [Williamson, 2009, Willimon et al., 2010] use a robot to manipulate soft objects, segment them and build models using a skeleton based approach. While these methods provide a means to manipulate, grasp and learn about objects, there are limiting assumptions made about the environment. The background is necessarily plain and prior knowledge is needed to infer the location of the object in order to grasp it.

2.3 Observations and Relation to the Hypothesis

The literature review gives rise to a number of gaps and potential avenues for further research. The work presented throughout the thesis focuses on a robot's ability to perceive and learn, both of which are explored in the literature concerning robotic object recognition and classification.

The work on robotic segmentation allows a robot to distinguish an object from its background by pushing it. These algorithms differ from a generic segmentation algorithm since they can both;

- **localise**, or determine the object it is playing with from other objects in the environment;

- and **segment**, or determine the boundary of the object that is being manipulated.

The combination of these two properties already shows the benefit of using proprioception for use in robotic perception. Computer vision algorithms can separate an image into regions likely to contain distinct objects, but are not able to determine which object is being manipulated.

The gap lies in the use of image subtraction in robotic segmentation. Current algorithms in the area use subtraction to determine areas of the image that have moved. This limits the algorithm's ability to perform when other objects in the environment are moving. This motivates the use of robotic motion to aid in segmentation. This idea is explored in Part II, showing an improved quality of segmentation when objects in the background are moving.

Once the background data has been filtered out, a model of the object can be built. Robotic learning algorithms use some form of background removal to isolate objects in the workspace. There are two main observations on the work in this area compared to learning algorithms in computer vision:

- **The algorithms are not online**, this means that a robot is not able to update its model as new data becomes available.
- **Robotic segmentation is not used** to filter the background. The methods either assume a plain background or use some prior knowledge to find the object. For example, [Li and Kleeman, 2011] assume that objects are symmetric in order to segment.

The work throughout III explores online learning on a robotic platform, using Latent Dirichlet Allocation. This work does not use segmentation and assumes a background with a low-density of features. Chapter 8 goes on to combine segmentation and learning to create a system which can autonomously learn robust object models in cluttered environments.

Referring back to Table 2.1, the literature is separated into *vision* and *robotics*, with the former meaning the computer has no control over the environment, the latter meaning it does. It is also separated into *segmentation* and *classification*; segmentation meaning to break an image into disjoint regions, and, classification meaning to identify an object's class from an image

of it. The second part of the thesis is dedicated to segmentation, and fits into the robotic segmentation area. The work presented here is comparable to three other techniques, but delivers an improvement when the background is changing. The third part to the thesis looks at robotic classification. The work presented here introduces self-supervision and topic modelling to the literature, and also goes on to combine segmentation and learning in the final system.

Chapter 3

A Case Study in Vision and Manipulation

The case study presented in this chapter introduces a new method for combining robot dynamics and vision to perform hand-eye coordination, and shows it to be successful at intercepting a moving object. The work does not directly confirm the hypothesis, but shows that visual information can be combined with robotic sensor data in order to benefit motion generation. The ability to generate trajectories is fundamental to robotics and is required throughout the rest of the thesis.

Current methods for hand-eye coordination generate robot trajectories that are based on the position, velocity and acceleration of the arm, but do not consider the force needed to make the movement. In this work, a statistical model for the vision system is derived, along with a Lagrangian dynamical model of a robotic arm, showing how to relate joint torques to the vision system. Because of the Bayesian approach to integrating the different distributions and modelling the vision, the method is robust to noisy data when estimating the position of the object. It also allows a degree of flexibility to change prior information about the trajectories.

The method is tested in simulation to show that it can intercept a moving object, while remaining optimal to a set of criteria. The study includes energy minimisation and precision combined with visual accuracy to intercept the target object.

The first part of the chapter is concerned with finding a common space

through which all of the sensors can communicate. The space that is most conceptual to work with is three dimensional Euclidean space. This correlates well with the model that humans use to understand the world. Once a common world is established, making decisions to move through it is much easier. These decisions can then be influenced by the physical body of the robot. For instance, it may be beneficial for a robot to make decisions based on the amount of energy it needs to make a movement, or how accurately it can take a measurement.

In summary, the chapter is broken into the following sections,

- **Robot Calibration**. The robot is able to measure the position of its joints, but this does not directly relate to the 3D position of its end-effector. In this section the robot is calibrated, finding a function which maps joint positions to end-effector pose and position in 3D.

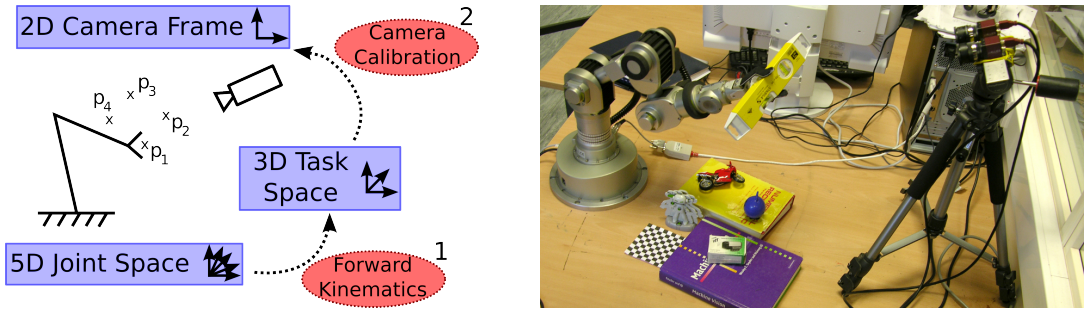
- **Camera Calibration**. There is also a complex relation between camera sensor data and 3 dimensional Euclidean geometry. Cameras measure in *projective* space, meaning that points further from the camera focus appear closer together than points measured near the camera. This relationship is modeled, finding a function which maps points in 3D to points in the camera space.

- **Trajectory Generation**. Once all of the different sensors can be measured in the same space, it is possible to define trajectories for the robot end-effector to move along. Statistical models of the vision system and robot are built in order to find optimal trajectories for the task of intercepting a moving target.

Solving these problems provides the basis for a robot to manipulate objects in the environment, a foundation for work throughout the rest of the thesis.

3.1 Robot Calibration

The robotic manipulator has sensors to encode the position of each joint. This is useful for moving the motors independently, but does not give the position or pose of the end-effector. Given a position in 3 dimensions, the robot should be able to calculate the joint positions needed to reach that point, the



(a) Calibration.

(b) Experimental Set-up.

Figure 3-1: The robotic set-up and calibration process. The arm moves in front of the cameras, points on the end-effector are recorded in order to find a map between the camera system and the robot motors.

inverse-kinematics of the robot. Conversely, given the positions of each joint, the robot should be able to calculate the position of the end-effector, this problem is known as the *forward-kinematics*.

Knowing the kinematics is important for hand-eye coordination, since it maps the sensor data into the 3D space surrounding the robot. Other sensors, such as cameras, can then easily transfer information to and from the robot manipulator. In the context of hand-eye coordination, we would like to be able to visualise a trajectory for the robot to move along, and convert it to motor positions. The full calibration diagram is shown in Figure 3-1a. This section gives a brief overview of the kinematic computations used throughout the rest of the thesis.

3.1.1 Kinematics

Finding the full kinematics of a robotic arm can be complicated, particularly for robots with a large number of degrees of freedom. The problem is well studied in the robotics community and too extensive to review here; solutions exist ranging from analytical methods, which are inexpensive to compute but complicated to solve; to numerical methods, which are computationally slower but can deal with much higher degrees of freedom. The methods used in this chapter are taken from [Zhao and Badler, 1994, Spong and Vidyasagar, 2008].

Forward kinematics is the problem of calculating the euclidean position $\boldsymbol{x}$ of an N degree of freedom robot arm, given the joint angles $\boldsymbol{\theta}$, that is,

calculating a map,

$$\boldsymbol{f} : \mathbb{R}^N \rightarrow \mathbb{R}^M \tag{3.1}$$

$$\boldsymbol{\theta} \mapsto \boldsymbol{x} \tag{3.2}$$

Where M is the number of dimensions in Euclidean space and also the degrees of freedom for the pose of the end-effector. For example, the robot may have 6 degrees of freedom, with an end-effector which can be rotated in 2 dimensions. In this case $N = 6$ and $M = 5 = 3 + 2$. Standard trigonometry is used to compute this function, and the details of this computation can be found in [Spong and Vidyasagar, 2008]. The positions of each link can be computed independently and summed to find the end-effector.

If velocity information can be read from the joint sensors then the velocity of the end-effector can be found by differentiating $\boldsymbol{f}$, using the multivariate chain rule,

$$J^{\boldsymbol{f}}_{\boldsymbol{\theta}} \dot{\boldsymbol{\theta}} = \dot{\boldsymbol{x}}. \tag{3.3}$$

Finding the **inverse kinematics** poses a much more difficult problem, particularly for an analytical solution. Analogously to Equation 3.2 an inverse map needs to be found,

$$\boldsymbol{f}^{-1} : \mathbb{R}^M \rightarrow \mathbb{R}^N \tag{3.4}$$

$$\boldsymbol{x} \mapsto \boldsymbol{\theta} \tag{3.5}$$

The position of the end-effector usually has fewer dimensions than the number of joints, and so the problem of finding an appropriate map is under-constrained. In general there is more than one solution to this problem. It is possible for there to be redundant solutions which are reachable by the robot, Figure 3-2 shows how a single point can be reached in two configurations of a 2D two link robot. Extending the problem in to 3D, potentially leads to an infinite number of reachable solutions.

The simplest solution is to use a non-linear optimisation method, choosing the robot's current configuration as a starting point. For a fixed position $\boldsymbol{x}$, an

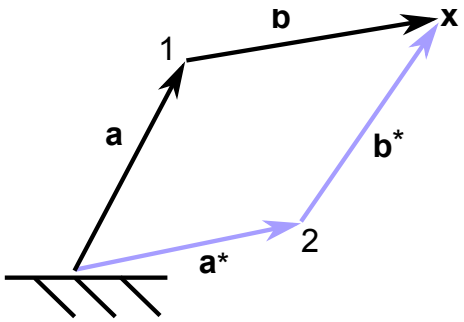


Figure 3-2: A diagram showing how a 2 link robot can reach multiple solutions. The manipulator is set to reach a point $\boldsymbol{x}$, a pair of vectors can be constructed to reach $\boldsymbol{x}$ in two different configurations, i.e. $\boldsymbol{a} + \boldsymbol{b} = \boldsymbol{x}$ and $\boldsymbol{a}^* + \boldsymbol{b}^* = \boldsymbol{x}$ with $|\boldsymbol{a}| = |\boldsymbol{a}^*|$ and $|\boldsymbol{b}| = |\boldsymbol{b}^*|$.

objective function is defined,

$$\boldsymbol{g}(\boldsymbol{\theta}) = \boldsymbol{f}(\boldsymbol{\theta}) - \boldsymbol{x} \tag{3.6}$$

Extra constraints can be formulated (such as lower and upper bounds on some of the angles). The starting point is taken to be the current position of the robot. This provides enough information for Newton's algorithm [Coleman et al., 1992] to effectively find a solution.

The solution used throughout the rest of this thesis is analytical, making it faster to compute. Details of the method can be found in Appendix A.1.

3.2 Camera Calibration

A camera is a sensor which takes points in three dimensions and projects them onto a plane. When observing a scene through a camera, points which are further away appear to be closer together than points which are near the camera focus. This is known as *perspective*, and does not conform to standard 3D Euclidean geometry.

Modelling a camera mathematically, involves building a model of perspective and then estimating a map between the world the cameras live in, and the 3D Euclidean geometry. This section explains the type of geometry capable of modelling a standard perspective camera, projective geometry, and the process required to estimate its parameters.

3.2.1 Projective Geometry

To build an accurate model of a camera while maintaining simplicity, standard Euclidean geometry is not enough. In the world of Euclid, parallel lines never meet. When viewing lines through a camera, train tracks for instance, parallel lines meet at a point at infinity. *Projective geometry* deals with this problem, defining points at infinity with mathematical rigour.

In Euclidean geometry each point is a vector. Consider a plane for instance, in this space parallel lines do not meet. Mathematically speaking, consider the parallel lines $\boldsymbol{\alpha} \cdot \boldsymbol{p}_1 = c_1$ and $\boldsymbol{\alpha} \cdot \boldsymbol{p}_2 = c_2$, with the constant gradient $\boldsymbol{\alpha} \in \mathbb{R}^2$, variables $\boldsymbol{p}_1, \boldsymbol{p}_2 \in \mathbb{R}^2$ and $c_1, c_2 \in \mathbb{R}$ with $c_1 \neq c_2$. The lines cross when $\boldsymbol{p}_1 = \boldsymbol{p}_2$ but this then implies that $c_1 = c_2$, a contradiction.

In projective geometry an extra dimension is added to the euclidean space, this is known as *projective space* denoted $\mathbb{RP}^n$ for n dimensions. A 2 dimensional point in euclidean space $(p_1, p_2) \in \mathbb{R}^2$ is equivalent to the point $(p_1, p_2, 1) \in \mathbb{RP}^2$, but in projective space there is an equivalence class of points which are considered equal, in fact $(p_1, p_2, 1) = (kp_1, kp_2, k)$ for and $k \in \mathbb{R} \setminus \{0\}$. If an arbitrary point is taken from $\mathbb{RP}^2$ dividing by the last element gets us back to $\mathbb{R}^2$. This makes the two spaces almost equivalent, except there is a set of points in $\mathbb{RP}^2$ which can't be mapped back to $\mathbb{R}^2$. Take $(p_1, p_2, 0) \in \mathbb{RP}^2$ the corresponding point in $\mathbb{R}^2$ is $(p_1/0, p_2/0)$, which does not really exist. This set of points is known as the *line at infinity*.

If we now consider a pair of lines in projective space, as before with $\begin{pmatrix} \boldsymbol{\alpha} \\ c_1 \end{pmatrix} \cdot \boldsymbol{p}_1 = 0$ and $\begin{pmatrix} \boldsymbol{\alpha} \\ c_2 \end{pmatrix} \cdot \boldsymbol{p}_2 = 0$, and variables $\boldsymbol{p}_1, \boldsymbol{p}_2 \in \mathbb{RP}^2$. The lines meet when $\boldsymbol{p}_1 = \boldsymbol{p}_2$, this either implies that $c_1 = c_2$ but that breaks our assumption, or that the third element of $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ is zero i.e. they are both at infinity.

3.2.2 The perspective camera

The projective geometry introduced above gives the right tools to properly model a perspective camera. In its most general form it is a matrix $P \in \mathbb{R}^{3\times4}$ which takes points in $\mathbb{RP}^3$ and maps them into $\mathbb{RP}^2$.

Figure 3-3 is a diagram showing a perspective camera. Points in the three dimensional euclidean world are mapped onto the two dimensional camera

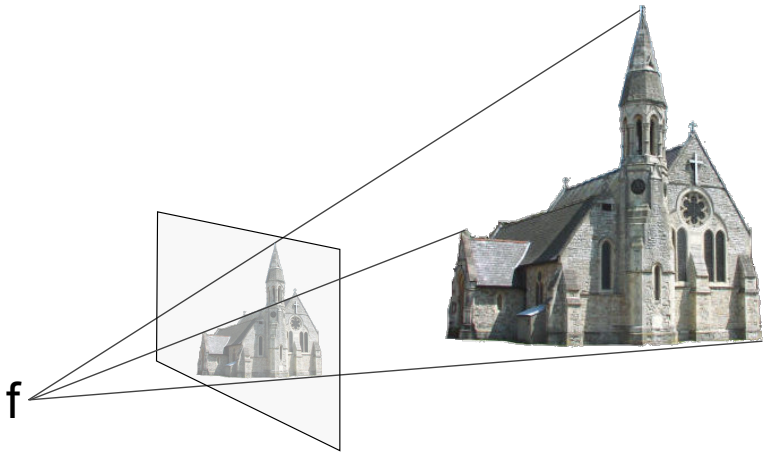


Figure 3-3: The perspective camera. Points in the 3 dimensional environment are mapped back on to the camera. Objects which are further away from the focus point appear smaller in the camera.

plane. In order to calculate a 3D position from a pair of cameras, matching and triangulation are needed, this process is explained in Section 3.4.1.

3.2.3 Estimating the camera matrix

Both projective geometry and a matrix camera model have been covered in previous sections, it remains to show how to estimate the parameters of the model. This is already a well known problem, but can be solved differently when using a robotic manipulator.

Estimating the projection matrix using computer vision alone is known as camera calibration [Hartley and Zisserman, 2000]. General methods exist to calibrate using only a pair of cameras and images of a chessboard (an object with clearly visible points of known *relative* dimension). Although these techniques can work very well, they put the cameras into an arbitrarily placed 3D coordinate frame with arbitrary units along each dimension. Performing calibration in this way requires a further affine map relating camera coordinates into robot coordinates. Instead, the cameras can be directly calibrated to the robotic arm, immediately placing them into the same 3D coordinate frame.

To estimate the matrix P, the robot is sent to a set of points in its work space. To make the estimation as accurate as possible these set of points should be as wide and as dense as possible. At each point, the corresponding position

in each camera is recorded. This can either be done manually or by putting a brightly coloured object on the end-effector of the robot and tracking it using a suitable algorithm, for example CAMShift [Bradski and Kaehler, 2008].

Assuming we have a set of known points in three dimensions $\boldsymbol{X}_i \in \mathbb{RP}^3$ and their corresponding point in the camera $\boldsymbol{x}_i \in \mathbb{RP}^2$. The projection matrix satisfies the following equation,

$$\boldsymbol{x}_i = P\boldsymbol{X}_i \tag{3.7}$$

Respecting that the image points are in projective space, this leads to the following pair of simultaneous equations,

$$(\boldsymbol{x}_i)_1 = P_{(1,:)}\boldsymbol{X}_i / P_{(3,:)}\boldsymbol{X}_i \tag{3.8}$$

$$(\boldsymbol{x}_i)_2 = P_{(2,:)}\boldsymbol{X}_i / P_{(3,:)}\boldsymbol{X}_i \tag{3.9}$$

Where $(\boldsymbol{x}_i)_k$ denotes the k^{th} element of $\boldsymbol{x}_i$ and $P_{(m,:)}$ denotes the m^{th} row of P. Rearranging these equations gives a matrix equation,

$$\begin{bmatrix} -(\boldsymbol{x}_1)_1\boldsymbol{X}_1^T & \boldsymbol{0}^T & \boldsymbol{X}_1^T \\ \boldsymbol{0}^T & -(\boldsymbol{x}_1)_2\boldsymbol{X}_1^T & \boldsymbol{X}_1^T \\ -(\boldsymbol{x}_2)_1\boldsymbol{X}_2^T & \boldsymbol{0}^T & \boldsymbol{X}_2^T \\ \boldsymbol{0}^T & -(\boldsymbol{x}_2)_2\boldsymbol{X}_2^T & \boldsymbol{X}_2^T \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{pmatrix} P_{(1,:)}^T \\ P_{(2,:)}^T \\ P_{(3,:)}^T \end{pmatrix} = \boldsymbol{0} \tag{3.10}$$

Letting A represent the matrix on the left and $\boldsymbol{v}$ the vector on the right we have,

$$A\boldsymbol{v} = \boldsymbol{0} \tag{3.11}$$

The trivial solution to this problem is $\boldsymbol{v} = \boldsymbol{0}$ which is not useful for this purpose. In fact, the matrix A is singular and so there exists another solution to Equation 3.11. Performing singular value decomposition on the matrix A and choosing the singular vector with smallest corresponding singular value yields the correct, non-zero, solution. When A is chosen to be the matrix shown in Equation 3.10, the solution corresponds to the camera parameters.

All of the mathematics introduced in previous sections relate sensors into a

common coordinate frame. This provides the basis for a system which can easily make calculations based on multiple sensors. The rest of the chapter deals with generating trajectories to make the robot move, based on both vision and robot sensor data.

3.3 Combining Vision with Dynamics for Trajectory Generation

Hand-eye coordination is an important problem in robotics, enabling a robot equipped with a vision system to manipulate visually sensed objects. This is particularly important in places where the uncertainty in reaching a point is very high, to direct the robot towards minimising error between the current and final pose.

The problem can be broken down into two distinct areas static and timed hand-eye coordination. In the static case, the robot must be able to move towards a point which is static in the scene, for example the picking and placing of fixed objects within the task space. A robot equipped with timed hand-eye coordination can move to match a trajectory of points through time, for example, a robot picking up moving objects from a conveyor belt, or a robot designed to hit a ball.

A significant amount of research has been done in the area of static hand-eye coordination; the image Jacobian can be found, relating velocities in the joints of the robot to points in the image [Chaumette and Hutchinson, 2006], [Chaumette and Hutchinson, 2007]. In a partitioned approach [Corke and Hutchinson, 2001] several points are tracked on the end-effector, ensuring the manipulator reaches the correct pose, taking care of the redundant degrees of freedom in the kinematics. Timed hand-eye coordination, introduced in [Allen et al., 1993], presents a much more difficult problem.

Progress has been made towards a solution using the image Jacobian, extending it to match the velocity of a visual trajectory [Chaumette and Hutchinson, 2007]. Dynamical systems can also be used to generate timed trajectories which make an interception [Santos and Ferreira, 2009], while allowing the robot to learn natural grasping movements.

This is extended into the field of mobile robotics, to intercept a moving ball. Methods in this area use statistics to find a distribution of possible ball positions [Guerrero et al., 2008], [Santos and Lima, 1993]; and then using controllers to intercept it [Capparella et al., 2005]. Both static and timed coordination have been solved using kinematics, giving positions, velocities and accelerations for the robot to move to. It is also important for the robot to be given the torques needed to make the movement. Force and vision have been combined in this way to better estimate and control the robot's position [Baeten and Schutter, 2004], [Lippiello et al., 2007], and allow the robot to apply a force to an object while maintaining a particular pose [Nelson et al., 1995]. This gives the robot the ability to use vision to position the arm and force sensors to undertake tasks which require more precision. Throughout the rest of this chapter, we explore a method for timed hand-eye coordination that can generate trajectories which are optimal in terms of the vision and dynamics of the robot.

3.4 Visual Trajectory Generation

To make the robot move, position commands are sent to the motors. The process of visualising a scene and determining a path for the robot to follow is known as *trajectory generation*. This section describes a method to generate trajectories based on the dynamics and uncertainty in the vision system, providing the first step towards integrating robot motion and the vision system.

3.4.1 Triangulation

Calibrating the robot in three dimensions (Section 3.1) provides a model of the cameras and robot that allow them to operate in the same three dimensional space. This means we can choose a point in the workspace and determine the joint positions needed to make the robot move to that point, and also, where it lies in the camera system. The converse problem is to find matching points in the camera system and find the corresponding point in three dimensions, this is known as triangulation.

Given a point $\boldsymbol{x}_1 \in \Pi_1$ in camera 1, a matching point $\boldsymbol{x}_2 \in \Pi_2$ in camera 2, and the projection matrices P_1, P_2. The problem is to find the three

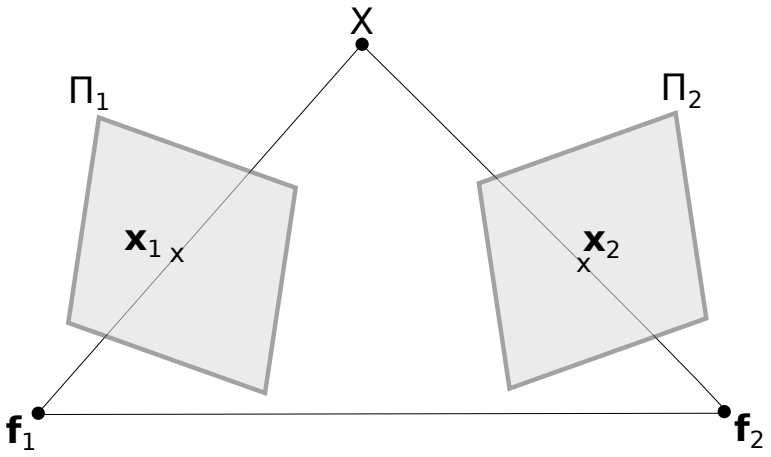


Figure 3-4: The epipolar geomtery of the cameras. A single point X in 3D is projected onto both of the cameras, Π_1 and Π_2. The reconstruction problem is to find the 3D point given the two corresponding camera points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$.

dimensional point $X \in \mathbb{RP}^3$ such that the following two equations hold,

$$\boldsymbol{x}_i = P_i X \ \forall i \in \{0, 1\} \tag{3.12}$$

There are many solutions to this problem, all with varying degrees of quality. The full process is detailed in [Hartley and Zisserman, 2000]. Figure 3-4 shows the *epipolar* geometry of the problem, and how the camera foci and matching 2D points relate to the corresponding 3D world point.

When the manipulator is constrained to a plane, such as a table top, the problem of mapping points into three dimensions can be simplified. Given a single camera matrix P, a point in the camera frame $\boldsymbol{x}$, the 3D world point X, and the parameters of the plane, the normal $\boldsymbol{n}$ and κ the following simultaneous equations are solved,

$$\boldsymbol{x} = PX \tag{3.13}$$

$$\kappa = \boldsymbol{n} \cdot X \tag{3.14}$$

The equations are then rearranged to give,

$$(x_1 P_{(3,:)} - P_{(1,:)})X = 0 \tag{3.15}$$

$$(x_2 P_{(3,:)} - P_{(2,:)})X = 0 \tag{3.16}$$

$$(\boldsymbol{n}^T - (0,0,0,1)\kappa)X = 0 \tag{3.17}$$

This system of equations can be solved using singular value decomposition, in a similar way to standard triangulation and camera calibration techniques. Further refinement using an optimisation algorithm is possible but beyond requirements for a robot which needs to operate quickly.

3.4.2 Visual Uncertainty

To build a model of the uncertainty in the vision system we assume that the error in measuring a point in each camera is Gaussian distributed. The problem is to then find out how the error is propagated through the triangulation, this gives the points in the 3D environment that are most accurate.

The epipolar geometry shown in Figure 3-4 motivates the problem. Consider two rays extending from the focus of each camera, through the lens. When the rays are close to perpendicular, a small deviation in each camera will not affect the calculation of X much. If the intersection is far away from the cameras then a small deviation in each camera could cause a large error in calculating the resulting 3D point.

Given the matching points in each camera, and the projection matrices, the probability of X is sought. Bayes theorem and the conditional independence of readings in each of the cameras gives,

$$p(X|\boldsymbol{x}_1, \boldsymbol{x}_2) = \frac{p(\boldsymbol{x}_1, \boldsymbol{x}_2|X)p(X)}{p(\boldsymbol{x}_1, \boldsymbol{x}_2)} \tag{3.18}$$

$$= \frac{p(\boldsymbol{x}_1|X)p(\boldsymbol{x}_2|X)p(X)}{p(\boldsymbol{x}_1, \boldsymbol{x}_2)} \tag{3.19}$$

Assuming that the likelihood of observing a 3D point is evenly distributed everywhere, they are possible to be observed by the cameras, the following

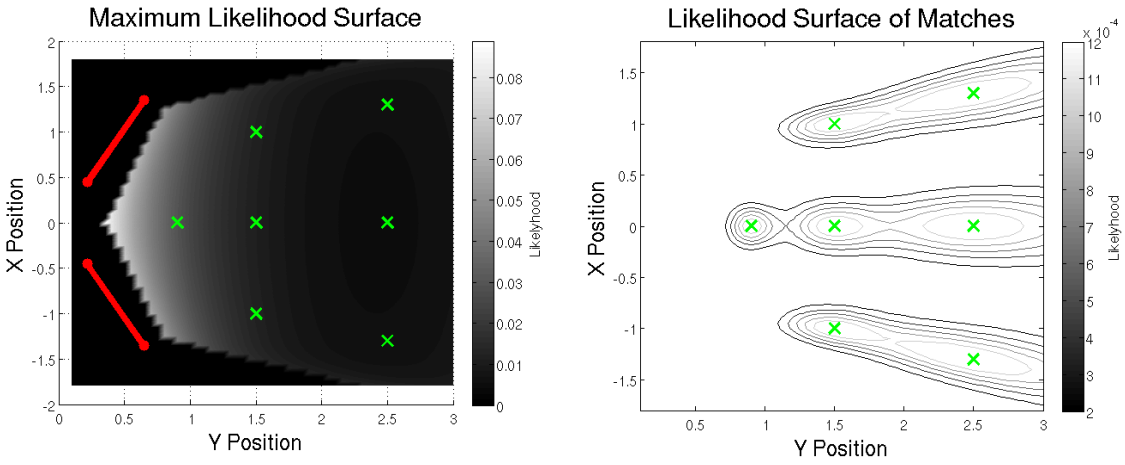


Figure 3-5: The accuracy in the vision system. Points which are further away from the camera focii reconstruct with much less certainty.

relation can be made,

$$p(X|\boldsymbol{x}_1, \boldsymbol{x}_2) \propto p(\boldsymbol{x}_1|X)p(\boldsymbol{x}_2|X) \tag{3.20}$$

The two distributions on the right hand side of Equation 3.20 are assumed to be normally distributed,

$$p(\boldsymbol{x}_i|X) = N(\pi_2(\boldsymbol{x}_i)|\pi_3(P_iX), \Sigma). \tag{3.21}$$

Where $\pi_n : \mathbb{RP}^n \to \mathbb{R}^n$ is the projection from real projective space into real space, representing a division by, and removal of, the last element of the input vector. The diagrams in Figure 3-5 represents the resulting probability map in 3D projected on to two dimensions.

When tracking a moving object we would like to be able to predict how it will move in the future. Doing this will allow enough time to make a decision and perform an action, before the object reaches a particular state.

In making this kind of prediction, the uncertainty grows with every time step taken without new data. Assuming that the object follows linear dynamics, the movement is modelled using a Kalman filter. This gives an estimated position and velocity for the object, and allows us to predict its state arbitrarily into the future.

A covariance matrix representing the corresponding measure of uncertainty

in estimating the state is also calculated, complementary to the probabilistic model used so far.

For each measured state $\boldsymbol{s}_t = \begin{pmatrix} z_t \\ v_t \end{pmatrix}$ at discrete time $t \in \mathbb{N}$, where z_t is the position and v_t is the velocity, the following relation holds,

$$\boldsymbol{s}_t = A\boldsymbol{s}_{t-1} + \varepsilon_t, \tag{3.22}$$

where A is the state-transition matrix and $\varepsilon_t \sim N(0, Q)$ is zero mean Gaussian noise. Letting $H_{(i,j)} = [\boldsymbol{s}_i, \boldsymbol{s}_{i+1}, ..., \boldsymbol{s}_j]$, the following matrix equation is solved finding a least-squares solution for A.

$$AH_{(1,N-1)} = H_{(2,N)} \tag{3.23}$$

$$A = H_{(2,N)}(H_{(1,N-1)})^*. \tag{3.24}$$

Where $(H_{(1,N-1)})^*$ is the Moore-Penrose pseudo inverse of the matrix $H_{(1,N-1)}$. This requires that N states are measured before any prediction can take place.

For known states the position and variance is estimated using the predict-update equations of a Kalman filter, for future time intervals the position is predicted using equation 3.22, and covariance predicted as follows,

$$M_t = AM_{t-1}A^T + Q. \tag{3.25}$$

This ensures that the covariance increases appropriately with the amount of prediction.

To incorporate this information into the final optimisation, a probability distribution $p(\boldsymbol{x}|t, K)$ is required where $t \in \mathbb{N}$ is the time and $K := H_{(1,N)}$ denotes the set of past measurements.

The probability distribution is given as follows,

$$p(\boldsymbol{x}|t, K) = p(\boldsymbol{x}|t, H_{(1,N)}). \tag{3.26}$$

$$= N(\boldsymbol{x}|\boldsymbol{s}_t, M_t) \tag{3.27}$$

Here $\boldsymbol{x}$ denotes the true position of the target and $\boldsymbol{s}_t$ the measured state at

time t. This distribution incorporates knowledge about the most accurate places in the vision system to catch the object, and also the best point in time.

3.4.3 Combining geometric and predictive uncertainty

We now have a probabilistic model governing the geometric relationship of the camera system, showing how the error in measuring a point is distributed in three dimensions (Equation (3-5)). A Kalman filter allows us to measure the position of an object and predict its future position and error distribution (equation (3.26)). It remains to show how these two distributions can be combined, to predict the position of a moving object while triangulating its position using multiple cameras.

Using the independence of the camera C and the Kalman filter with time (K, t), the likelihood of the position $\boldsymbol{x}$, given data from the tracked target $\beta := (C, K)$ is calculated.

$$p(\boldsymbol{x}|\beta, t) = p(\boldsymbol{x}|C, K, t) \tag{3.28}$$

$$\propto p(\boldsymbol{x}|C)p(\boldsymbol{x}|K, t) \tag{3.29}$$

This probability distribution gives a complete likelihood of the estimated position of a moving object, also allowing for future positions to be predicted. The formulation is general enough for alternative tools to be used if needed. For example, where the assumption of linear dynamics in the Kalman filter may be too limiting in some circumstances, it is possible to be easily exchanged with another particle filter which also gives information about the uncertainty in prediction.

3.5 Arm Dynamics

There are very large number of movements a robot can make in order to achieve a particular task. In this section we analyse a set of trajectories by considering the inverse dynamics of the robot, i.e. the torques needed to make the movement. This is then used as a basis to decide on which trajectory is the best.

To determine the inverse dynamics, a Katana robot is modeled in the program DySim [Sahinkaya, 2004]. Katana is a 6DOF small sized robotic manipulator, with a reach of 517mm and weighing 5.2kg.

DySim uses the Lagrange equation, considering the difference between kinetic and potential energy in the system to find the torque in each joint needed to make a given movement.

3.5.1 The Set of Trajectories

A small set of trajectories are selected to analyse on the robot. Bezier curves are used to make the movement, but with the additional constraint that the beginning and end of the trajectory have zero velocity and acceleration. This prevents the controllers from putting too much force on to the motors, making them stall or break.

A Bezier curve is defined by the following equation,

$$\boldsymbol{b}(t) = (1-t)^2\boldsymbol{p}_0 + 2(1-t)t\boldsymbol{p}_1 + t^2\boldsymbol{p}_2 \qquad t \in [0,1] \tag{3.30}$$

Then $\boldsymbol{b}(t)$ passes through the points $\boldsymbol{p}_0$ and $\boldsymbol{p}_2$ with curvature controlled by the point $\boldsymbol{p}_1$. To ensure that the trajectory terminates with zero velocity and acceleration a shaping function $f : \mathbb{R}^+ \to [0,1]$ is used [Sahinkaya, 2001], defined by the following equations,

$$f(t) = 1 - e^{-(\alpha t)^3} \tag{3.31}$$

$$\dot{f}(t) = 3(\alpha t)^2 e^{-(\alpha t)^3} \tag{3.32}$$

$$\ddot{f}(t) = (6\alpha t - 9(\alpha t)^4) e^{-(\alpha t)^3} \tag{3.33}$$

The parameter α controls the magnitude of acceleration. Setting $\alpha := 1.66$ ensures that the movement stops near $t = 1$.

The Bezier curve is then reparametrised by the variable t, to give a new

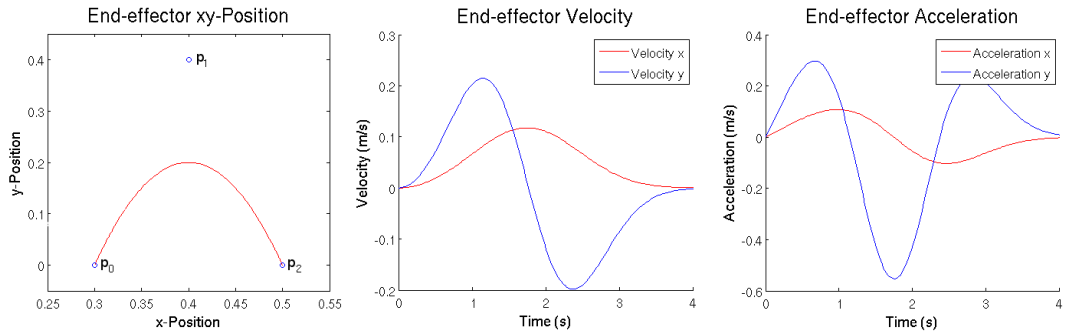


Figure 3-6: The position, velocity and acceleration are shown for a 2-dimensional Bezier curve, that has been modified by shaping the time input. The graphs demonstrate that in each dimension the trajectory has zero velocity and acceleration at the end points.

function $\Phi : \mathbb{R} \to \mathbb{R}^n$, defined by the following,

$$\Phi(t) = \boldsymbol{b}(f(t)) \tag{3.34}$$

$$\dot{\Phi}(t) = \dot{\boldsymbol{b}}(f(t))\dot{f}(t) \tag{3.35}$$

$$\ddot{\Phi}(t) = \ddot{f}(t)\dot{\boldsymbol{b}}(f(t)) + \ddot{\boldsymbol{b}}(f(t))\dot{f}^2(t) \tag{3.36}$$

Substituting (3.31) - (3.33) into (3.34) - (3.36) gives a smooth function with the correct boundary conditions, i.e. zero velocity and acceleration when $t = 0$ and $t = 1$.

A set of trajectories is then defined by the vector $\psi \in \Psi$ with,

$$\psi = (\theta, l, \gamma)^T \tag{3.37}$$

This relates to the Bezier curve defined by the points $\boldsymbol{p}_0$, $\boldsymbol{p}_1$, $\boldsymbol{p}_2$, as follows,

$$\boldsymbol{p}_0 = (0, 0.7, 0)^T \tag{3.38}$$

$$\boldsymbol{p}_2 = l(\sin(\theta), \cos(\theta), 0)^T \tag{3.39}$$

$$\boldsymbol{p}_1 = \boldsymbol{p}_0 + \frac{1}{2}(\boldsymbol{p}_2 - \boldsymbol{p}_0) + \gamma \boldsymbol{p}_2^{\perp} \tag{3.40}$$

Where $\boldsymbol{p}_2^{\perp}$ is the vector perpendicular to $\boldsymbol{p}_2$. Each of the parameters are shown in Figure 3-7.

This constrains the manipulator to move in the plane, to simplify degrees of

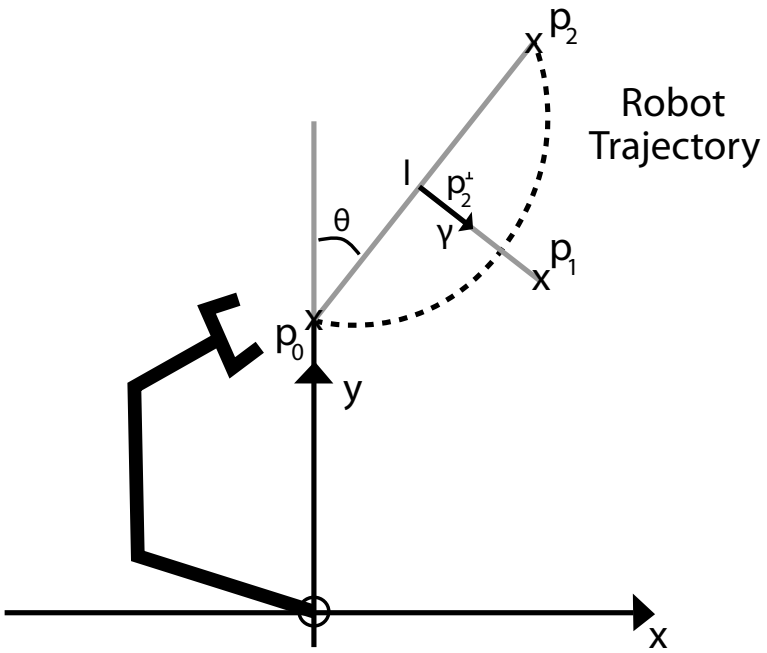


Figure 3-7: The geometry of the robot trajectories. The robot is constrained to a 2 dimensional environment, i.e. the last element of $\boldsymbol{p}_1$, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ are zero. The robot moves along a Bezier curve from $\boldsymbol{p}_0$ to $\boldsymbol{p}_2$. The parameter θ describes the angle the the robot begins moving in, l the distance it moves and γ how far the trajectory curves away from a straight line.

freedom in the model. With the parameter γ, giving the degree of freedom termed *curvature*. Figure (3-6) shows a Bezier curve along with the corresponding velocity and acceleration, demonstrating the zero terminal boundary conditions.

3.5.2 Energy Efficiency

We take the set of trajectories as defined in section 3.5.1 and calculate an energy cost function, more explicitly, the *mechanical work* [Mansfield and O'Sullivan, 2011]. This gives a good measure to determine one trajectory from another.

The average energy in the k^{th} joint is defined as follows,

$$E_k(\psi) = \int_{t=0}^{v} |f_k(t)|dt \tag{3.41}$$

This is the integral of the absolute sample mean torques over time, where

$f_k(t)$ denotes the torque in joint k at time $t \in [0, \upsilon)$. The total energy consumption for the action is taken to be the the sum over each of the joints.

To test that the calculation of energy is correct, the same trajectories are performed on the Katana robot and the total current needed to make the movement is measured. Figure 3-8 shows the results of this experiment for a range of trajectories, the simulated results are similar to those from the dynamic simulation, validating the model.

3.5.3 Precision

Another measure of quality between trajectories is precision. A trajectory is called precise if adding noise to any of the motor torques affects the final position of the end-effector the least. This is useful for robots which do not use a high gain for position feedback, an example of this would be a torque controlled robot which is designed to be compliant.

To calculate precision, random Gaussian noise is added to the desired end position of the robot. The trajectory needed to reach this point is then calculated. This process is repeated 500 times, giving a sample set of the torques required to make the move in each joint. Counter intuitively, precise trajectories are chosen to be ones which have the lowest signal to noise ratio, ensuring that small perturbations in the motor torques will affect the end-position the least.

Two examples of this are shown in Figure 3-9, with each colour representing a different motor. The bottom set of torque trajectories have a lower signal to noise ratio and are therefore more precise.

The signal to noise ratio for the k^{th} joint is then given by,

$$\rho_k(\psi) = \int_{t=0}^{\upsilon} \frac{\overline{\mu}_k(t)}{\overline{\sigma}_k(t)} dt \tag{3.42}$$

Where $\overline{\mu}_k(t)$ and $\overline{\sigma}_k(t)$ are the mean and variance of the torques in the k^{th} joint at time t. ψ denotes the parameters of the trajectory.

This can then be used to estimate how the covariance of the end-effector

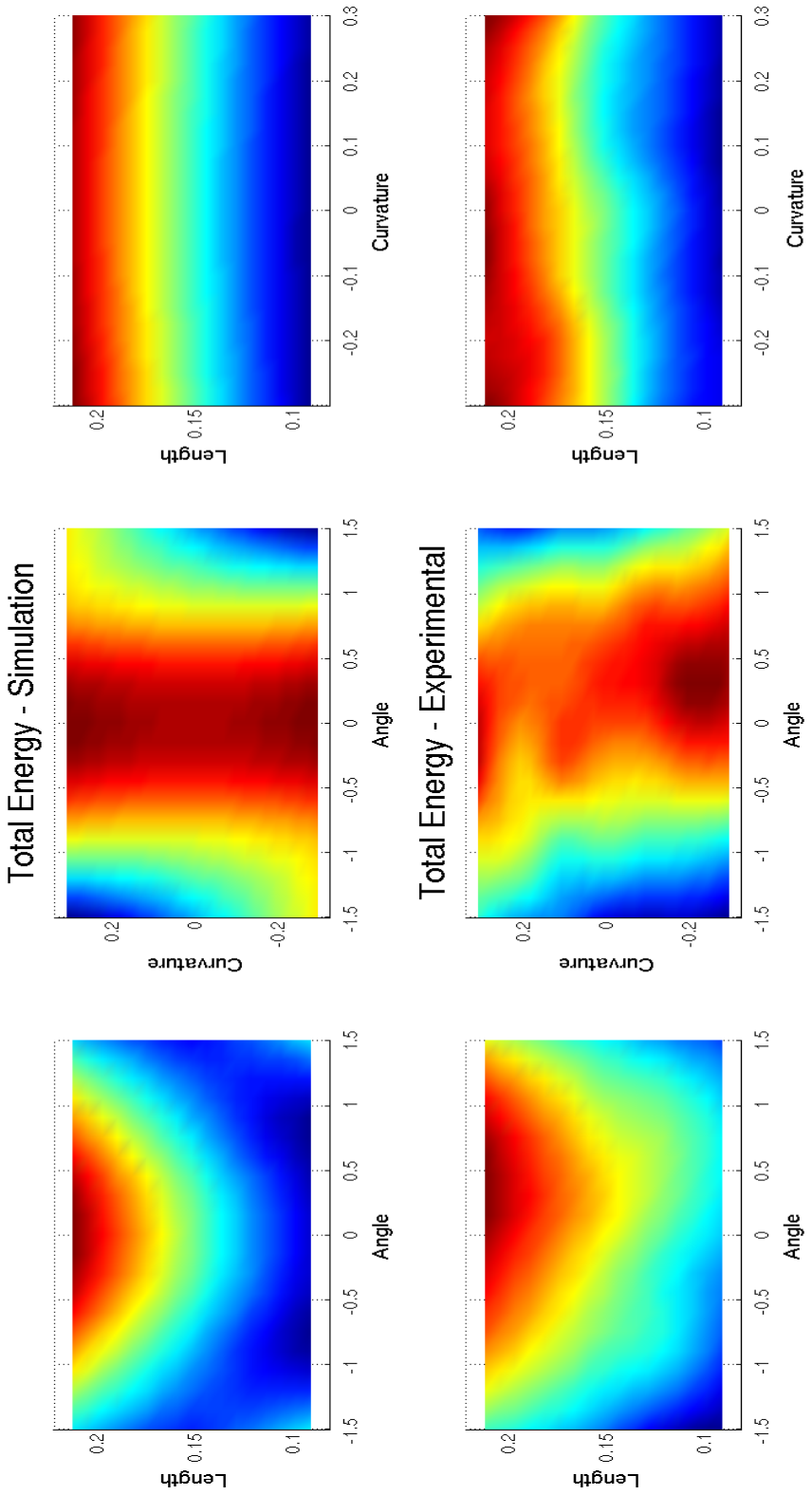


Figure 3-8: The total energy required to perform a particular trajectory. The top row shows the results from a dynamic simulation, and the bottom row shows the results of measuring the total current used in the motors of the Katana.

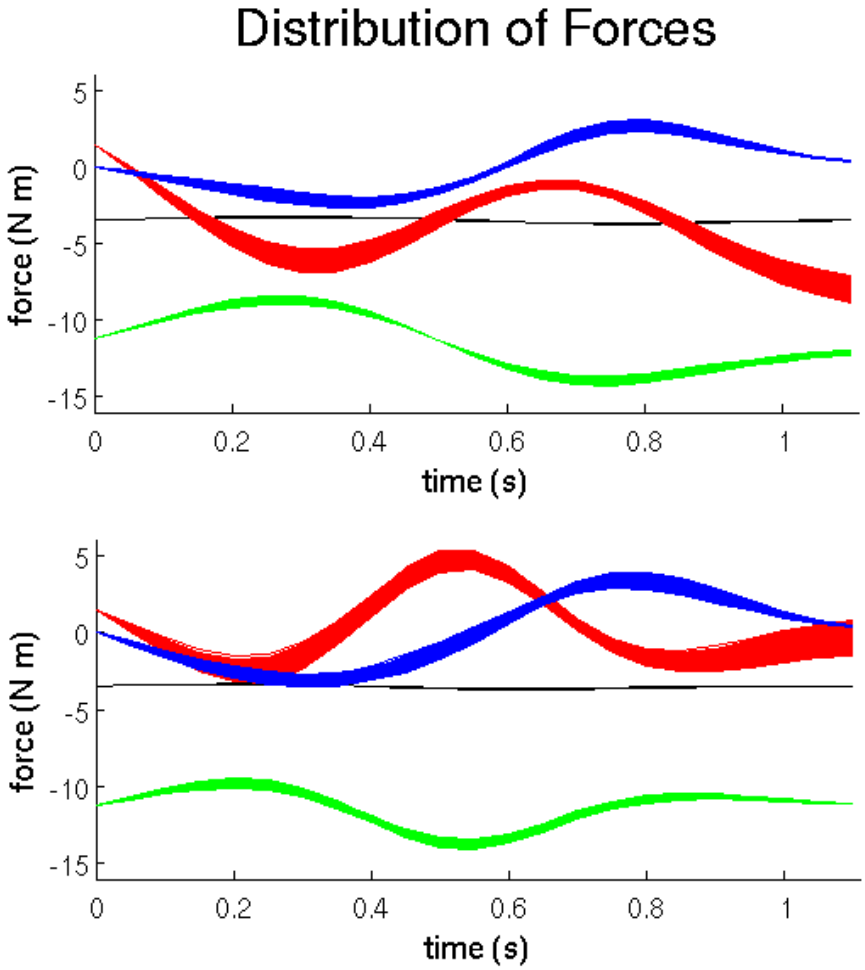


Figure 3-9: The resulting torques calculated from a distribution of end-effector positions. The top graph shows a the torques needed to generate a Bezier curve with parameters $\theta = 0$, $l = 0.15$ and $\gamma = 0$. In the bottom graph the trajectory has parameters $\theta = -1.5$, $l = 0.07$ and $\gamma = -0.3$. The graphs show each measured trajectory plotted one on top of the other.

position changes, given that there is noise present in the torques.

$$\Sigma_\psi \propto (\sum_{k=1}^{4} \rho_k(\psi))^{-1} I_3. \tag{3.43}$$

If a robot with high precision controllers are used, a small constant covariance matrix can be used instead.

3.6 Combining Vision and Dynamics

In this section, the arm dynamics and probabilistic models of the vision system are combined in order to perform hand-eye coordination.

Given the statistics for the arm α and object β being tracked, and the time passed $t \in \mathbb{R}^+$, a distribution over the set of trajectories can be found,

$$p(\psi|\alpha,\beta,t) \propto p(\alpha,\beta|\psi,t)p(\psi) \tag{3.44}$$

$$\propto \iint_{\boldsymbol{x}^b,\boldsymbol{x}^a} p(\alpha,\beta|\psi,\boldsymbol{x}^a,\boldsymbol{x}^b,t)p(\boldsymbol{x}^a,\boldsymbol{x}^b|t,\psi)p(\psi) \tag{3.45}$$

$$\propto \iint_{\boldsymbol{x}^b,\boldsymbol{x}^a} p(\alpha|\psi,\boldsymbol{x}^a,t)p(\beta|\boldsymbol{x}^b,t)p(\boldsymbol{x}^a,\boldsymbol{x}^b|t,\psi)p(\psi) \tag{3.46}$$

Where $\boldsymbol{x}^a$ and $\boldsymbol{x}^b$ denote the positions of the arm and object respectively, and ψ represents the parameters of the trajectory.

The Dirac-delta distribution is then used, so that a touch is only probable when the position of the object and position of the hand coincide.

$$p(\boldsymbol{x}^a,\boldsymbol{x}^b|t,\psi) = \delta(\boldsymbol{x}^a - \boldsymbol{x}^b) \tag{3.47}$$

Using Bayes theorem and equation (3.47) with equation (3.46) then gives that,

$$p(\psi|\alpha,\beta,t) \propto \int_{\boldsymbol{x}} p(\boldsymbol{x}|\alpha,t,\psi)p(\boldsymbol{x}|\beta,t)p(\psi) \tag{3.48}$$

$p(\boldsymbol{x}|\beta,t)$ is the distribution of error in the visual estimate of the moving object, given in equation (3.29), with β defined at the end of section 3.4.3; $p(\boldsymbol{x}|\alpha,t,\psi)$ is the distribution of error in the estimate of the arm position, taken to be normally distributed, with covariance as in equation (3.43) and a mean corresponding to the measured arm position at time t.

The distribution $p(\psi)$ is known as the prior, and gives the likelihood of a trajectory independent of any other information. The prior is set to be

proportional to the total energy required to make the movement,

$$p(\psi) \propto (\sum_{k=1}^{4} E_k(\psi))^{-1} \tag{3.49}$$

To find an action ψ the negative log-likelihood of (3.48), $-ln(p(\psi|\alpha, \beta, t)$ is optimised using the Gauss-Newton technique. The initial value is estimated by sampling the whole domain sparsely, and then choosing the maximum.

3.7 Simulations

The method is tested by dynamically simulating a Katana robot, and a linearly moving object in DySim. To ensure that the simulation environment is as accurate as possible, the same set of movements are performed on the robot as in simulation, and compare the total energy outputs using equation (3.41). The trajectories are defined by Bezier curves, as in section 3.5.1, with the following defining parameters, $l \in [0.07, 0.21]$, $\theta \in [-1.5, 1.5]$, $\gamma \in [-0.3, 3]$. Sampled to give a total of 252 trajectories.

In order to compute the total energy for the robot, the trajectories were performed 10 times, and the current required in each of the joints recorded and averaged, this gives a value roughly proportional to the torque. This mean current is then used as $f_k(t)$ in equation (3.41), to calculate the total energy exerted by the robot throughout the movement.

The validation of the dynamics of the robot allow us to move on to use only simulation for experimentation. This simplifies the amount of technology needed and allows us to concentrate on the functionality of the method instead.

The ability of the robot to intercept the target is shown in Figure 3-10. The method starts predicting when the trajectory becomes red, and the size of marker is proportional to the covariance, as estimated by the Kalman filter. This demonstrates how uncertainty in the object's position increases with the amount of prediction. The robot is optimising for both energy reduction and precision in the trajectory and, as a result, moves in a straight line towards the point of interception.

In order to show the effect of using different priors, the method is run and

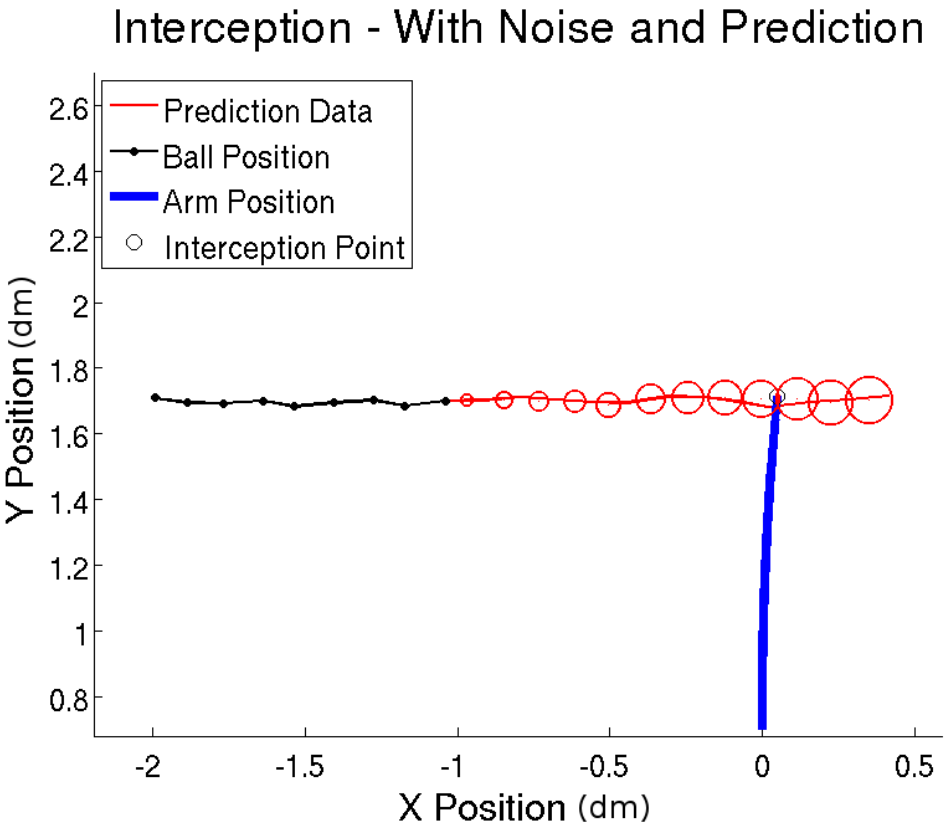


Figure 3-10: Intercepting a moving target, the uncertainty in prediction is represented proportionally to the marker size.

optimised for both energy minimisation and precision; energy minimisation alone with the end-effector covariance taken as 0.01; and precision alone with the prior $p(\psi)$ set to be uniformly distributed.

The magnitude of the noise is increased and, as the standard deviation increases, the method tends to choose interception points earlier (figure 3-11). Comparing this with Figure 3-10 it can be seen that as the variance increases, the robot starts to choose interception points earlier along the objects path, minimising the uncertainty in the Kalman Filter.

The response to noise was tested by adding Gaussian noise to the position of the moving object with a standard deviation between 0 and $3cm$, this was done for 50 trajectories and the average distance between the object and end-effector was consistently below $3mm$. This is just above the minimum tolerance of $2.5mm$, which is calculated as the average distance when zero noise is added to the objects position. As a result, if the target object has a diameter of $6cm$, then assuming that the tracker estimates its position within the boundary, the standard deviation will be less than $3cm$, allowing the object to be intercepted with a high probability.

3.8 Discussion

A restriction of the method is that it is open loop with respect to the cameras, so changing the calibration would affect the results. To remedy this, a visual feedback controller (such as defined in [Chaumette and Hutchinson, 2007]) can be used, after a decision has been made on the correct trajectory.

It would also be possible to extend the method so that a variety of priors can be combined. Considering the torques in the motors, the prior can be used selectively depending on the application. For example, it may be important to avoid certain areas of the scene that are blocked by objects, or are out of reach for the robot. Trajectories that exert the maximum amount of torque may be needed in certain areas of the scene, in order to pick up an object, or push something out of the way. These improvements are beyond the scope of the thesis, and would make a good candidate for future research.

The experiments show that, as the uncertainty increases, the robot tends to chose trajectories that intercept the object early on. Determining the optimal

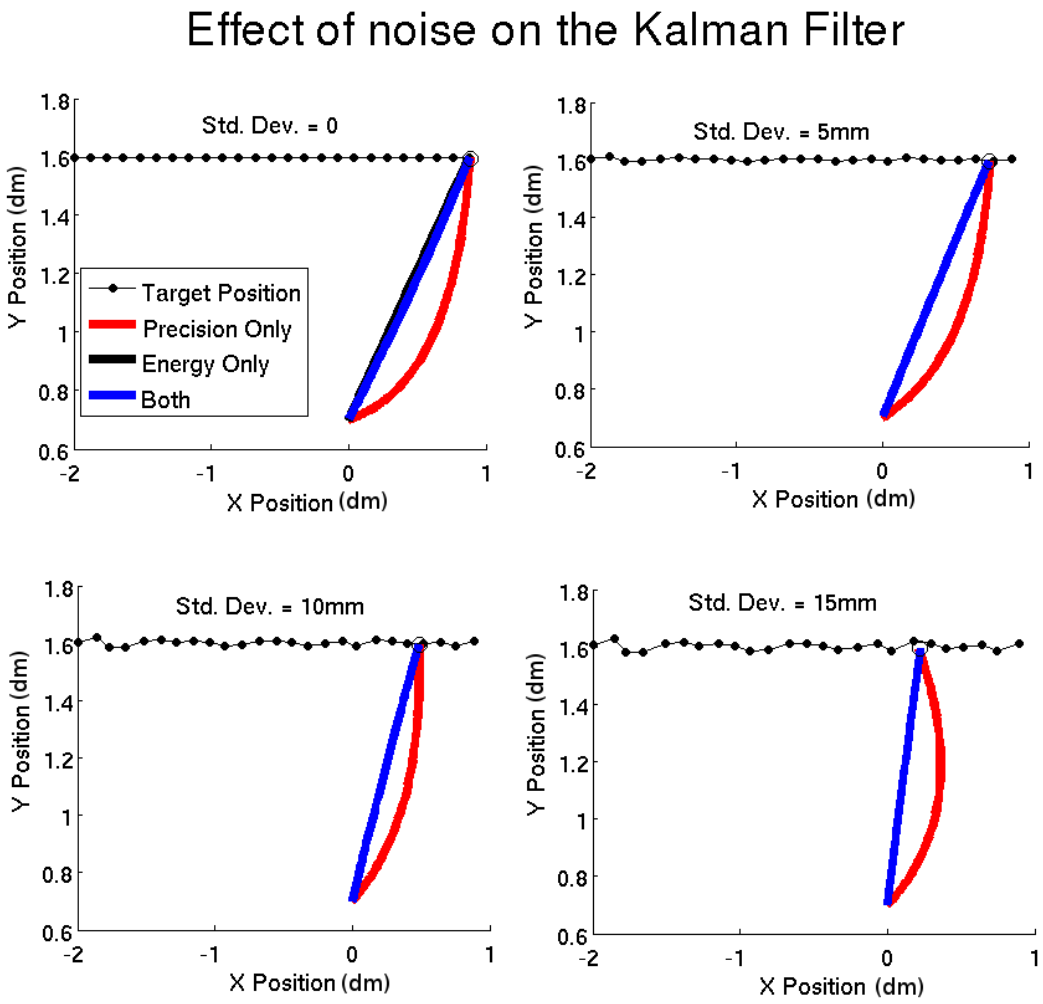


Figure 3-11: Different actions the robot takes under varying noise and priors. In the final three graphs the trajectories for energy efficiency and combined energy and precision coincide, this shows a prior with a stronger preference for a trajectory dominating the other. Both the 'Energy Only' (black) and 'Both' (blue) lines coincide for the whole trajectory.

time to start predicting is something that can't be calculated before making a decision. We plan to solve this problem using machine learning, enabling the robot to learn about the best prediction times incrementally, through intercepting the object several times.

This concludes the results and method for robotic / visual calibration and trajectory generation. The next part of the thesis describes a method to use the known position of the robotic arm, within the camera frame, in order to segment objects.

Part II

Robot Perception

Chapter 4

Object Segmentation

The previous part of the thesis introduces the tools needed to calibrate the robot and cameras together, and generate trajectories in the robot's workspace. The robot is then able to move to a set of points defined by the vision system. The method introduced in this part allows the robot to perceive objects using the known motion of the robotic arm. Perception is defined as bottom-up object localisation and segmentation of the object being manipulated.

The method is in support of the hypothesis, since it shows that robotic motion can be used to identify the object that the robot is interacting with. Although it does not require any prior knowledge, a prior can be used to further improve the result, allowing a top-down element to visual perception.

The main contributions of this work are to provide a method which:

- Proves that **robotic motion** can be used to **localise and segment** objects manipulated by the robot.
- Improves current robotic perception methods, allowing object segmentation in cluttered environments with **moving backgrounds**.
- Gives a segmentation that is **probabilistic**, providing a degree of certainty in any part of image being part of the object.

The chapter begins by introducing the theory behind the method, and some background surrounding bottom-up and object segmentation in robotics, before describing the technique in detail.

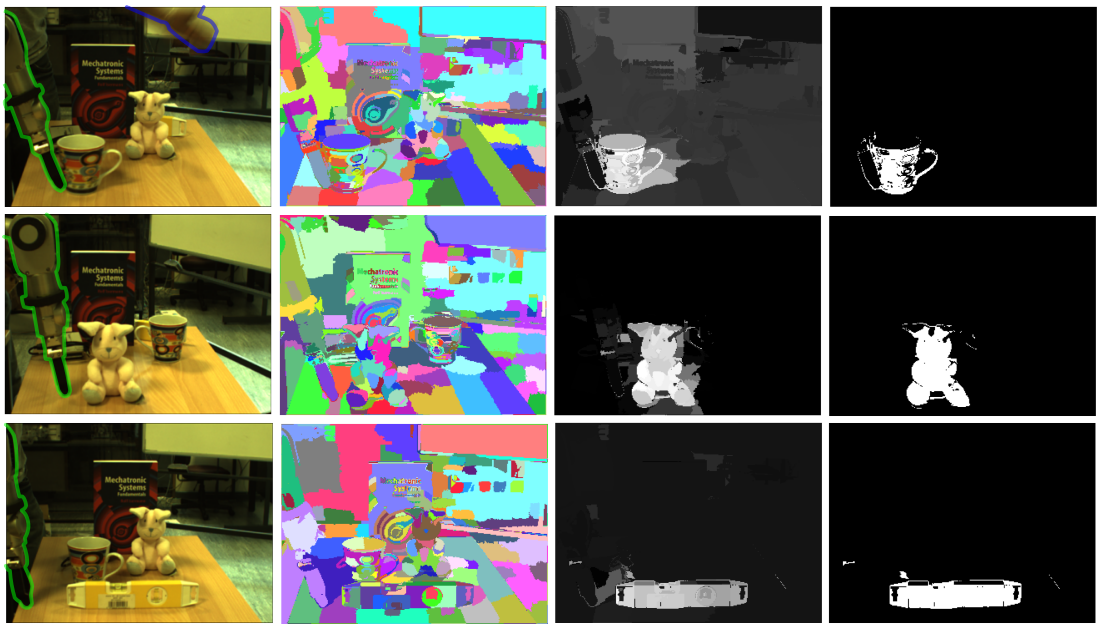


Figure 4-1: Results from the *Motion-Likelihood* method. From left to right: (1) A frame from the video. (2) The corresponding low-level segmentation. (3) The resulting probability map over each segment. (4) The thresholded probability map corresponding to the object of interest.

4.1 Method Overview

The method allows a robot to manipulate an object in front of it (e.g. push or grasp), and use the visual information to segment the object from its background. This is done by a probabilistic process that correlates robot and image motion.

The correlation between the motion on the video and the robot arm is evaluated using a statistical motion model. The main component of the motion model is the motion transformation function. This function is used to represent how things can move in the environment. For example a rigid body motion transformation (*i.e.* objects that can't be deformed) implies that the pixels within the object must 'move together'. Other motion transformation functions, such as elastic or fluidic could be used, but are beyond the scope of the thesis. The motion transformation function is used to calculate the error between the observed motion in the image and the expected one for a given arm motion.

The method presented here comprises five steps:

1. Low-level image segmentation
2. Computing the motion in the image using dense optic flow
3. Mapping end-effector motion into camera coordinates
4. Estimating the arm-segment motion correlation
5. Thresholding the estimation to define the object of interest

In step 1, a video stream is taken and segmented according to the colour information in each frame. This gives the robot knowledge of where the segments are, and tracks them throughout the video. Mean shift segmentation is used for this, and is detailed in Section 4.2.1. Step 2 is calculated using optical flow, detailed in Section 4.2.2. In Step 3 the robot joint angles are converted into 2D camera coordinates. The method for robot calibration was described in the previous chapter. Step 4, estimates the correlation of segment motions with that of the robot arm. The result is a probability that any segment belongs to the manipulated object. Finally, Step 5 chooses the segments with highest probability and defines the object of interest. Figure 4-2 illustrates a diagram of the method. The blocks illustrate the previous steps and the images some of their outputs. Steps 4 and 5 are described in Section 4.4.

The process to calculate motion models and correlate them with the arm motion is detailed in the rest of this section. This process is inspired by the work of [Torr, 1998, Torr and Zisserman, 1998] for general motion segmentation.

4.2 Low-level Image Processing

General low-level image processing is used to extract meaningful information from images and videos. Algorithms of this class are known as *low-level* as they do not use prior information about the scene. This is useful for work carried out throughout the chapter as it provides the tools needed to analyse the motion of the video, and also to break the image into regions contained within objects. Higher level analysis can then be used to group together regions which move in a coherent manner.

Two types of low-level image processing are used throughout the chapter:

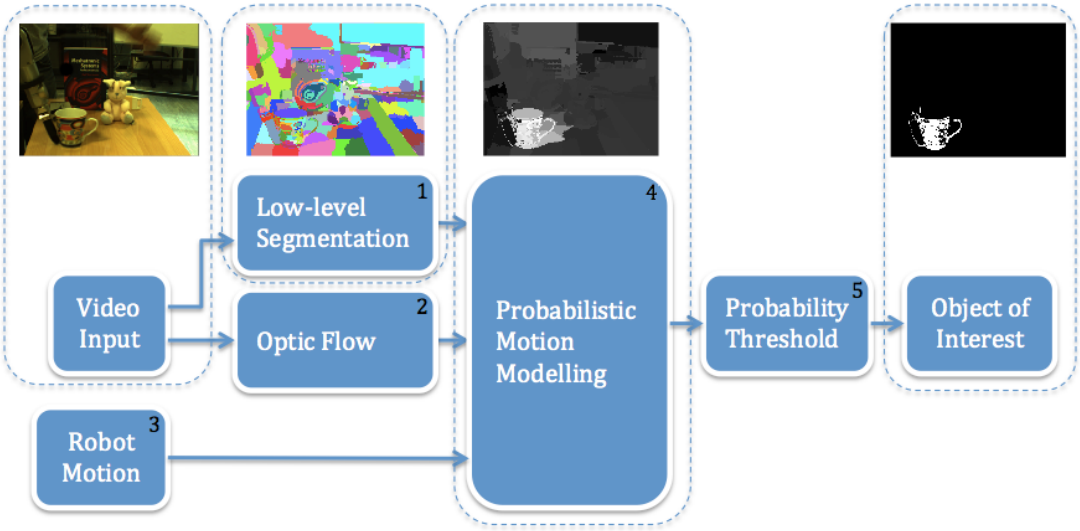


Figure 4-2: A diagram showing the full object segmentation process. The method takes input from a camera and the robot arm, the images are processed using low-level segmentation and optical flow. The probability that each segment matches the arm motion is calculated over a period of time. The resulting probability map is then thresholded to obtain a final object-background segmentation.

- **Video Segmentation**. A segmentation algorithm is used to separate a video into non-overlapping regions which contain pixels of a similar colour. The regions change continuously from frame to frame, so that parts of objects can be tracked through the video. The goal is to "over segment" the video, so that there is more than one region per object. This allows us to go on and merge regions which belong to the same object without the risk of including background.

- **Optical Flow**. Motion is extracted from the video using optical flow, which determines the direction that part of an object has moved from one frame to the next. This information is used to determine the overall motion of each segmented region, to correlate it with the robot arm motion.

There is a vast amount of literature surrounding these two problems. The work in this chapter uses two well known implementations which are described in more detail in the following subsections.

4.2.1 Mean Shift Video Segmentation

One of the most basic approaches to low-level video segmentation is Mean Shift [Cheng, 1995]. Mean shift is a clustering technique and in order to segment a video each pixel, $\boldsymbol{p}_i$, is represented by its colour, position and frame number, $\boldsymbol{p}_i = (r_i, g_i, b_i, x_i, y_i, f_i)$, respectively. A set of means $\boldsymbol{\mu}_k$ are then computed iteratively according to the following algorithm,

$$\boldsymbol{\mu}_{k+1} = \frac{\sum_{i=0}^{N} K(\boldsymbol{\mu}_k - \boldsymbol{p}_i)\boldsymbol{p}_i}{\sum_{i=0}^{N} K(\boldsymbol{\mu}_k - \boldsymbol{p}_i)} \tag{4.1}$$

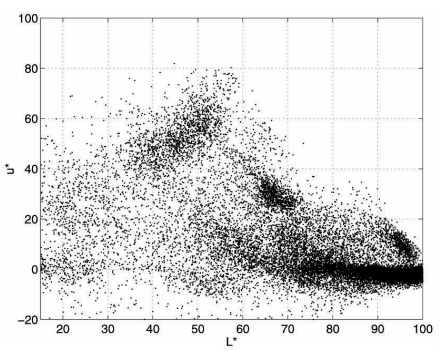
The function K can be any kernel, but is usually chosen to be Gaussian or flat. The algorithm is initiated at every point in the feature space, and re-cursed until $|\boldsymbol{\mu}_k - \boldsymbol{\mu}_{k+1}|$ becomes small enough. The points which converge to the same point are considered part of the same cluster. Figure 4-3 shows an example of the Mean Shift algorithm converging to the modes of 2D data, with a complex multi-modal distribution.

Using this algorithm for image segmentation was introduced in [Comaniciu and Meer, 2002]. Several improvements exist, including [Paris and Durand, 2007], who use Morse theory to optimise the speed of the algorithm. This method for segmentation is used throughout the chapter. It is simple and fast enough to operate, but is limited by its quality. Replacing the algorithm for one of higher quality could lead to an improvement in the final object level segmentation results, but is beyond the scope of the thesis. Instead topological mean shift is chosen, keeping speed of computation in mind.

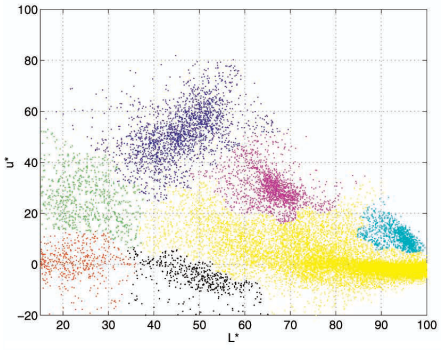
4.2.2 Optical Flow

Optical flow is the problem of determining how the scene, in front of the camera, moves throughout a video sequence. The only information available to the algorithm is the change of pixel intensity at each frame. Intensity information needs to be translated into a change of position in camera coordinates, representing a video as $I(\boldsymbol{x}, t)$, where $\boldsymbol{x}$ is the pixel position and t is the frame. Assuming that pixel intensities are translated from one frame to the next,

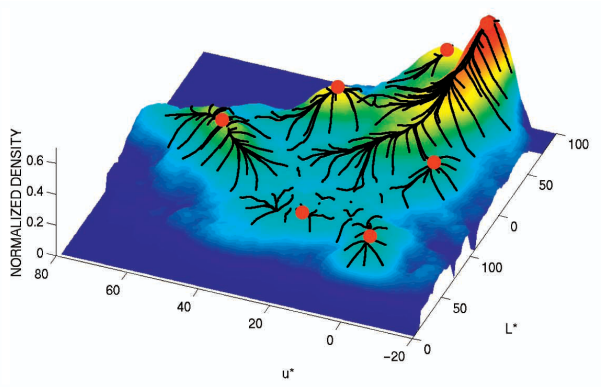
$$I(\boldsymbol{x} + \boldsymbol{u}, t + 1) = I(\boldsymbol{x}, t) \tag{4.2}$$



(a) 2 Dimensional set of feature points



(b) The data clustered according to Mean Shift



(c) Density function of the data, showing the convergence of points. The modes are shown in red.

Figure 4-3: Mean Shift clustering on 2D data. The subfigures show how a set of data is clustered using mean shift, and how start points around the distribution converge towards the local maxima. Image taken from [Comaniciu and Meer, 2002].

Where $\boldsymbol{u}$ is the velocity of the moving object. This constraint means finding a translation $\boldsymbol{u}$ which maps the pixel in the previous frame to the current frame. Using Taylors theorem and linearly approximating this constraint leads to,

$$J^I_{(\boldsymbol{x}^T,t)^T} \begin{pmatrix} \dot{\boldsymbol{x}} \\ 1 \end{pmatrix} = 0 \tag{4.3}$$

Where $J^I_{(\boldsymbol{x}^T,t)^T}$ is the Jacobian of I with respect to $(\boldsymbol{x}^T, t)^T$. The usual approach is to use an optimisation technique in order to find the velocities, given a video. This is covered in detail in [Horn and Schunck, 1981], and has been studied and extended throughout the computer vision literature. To date,

one of the most widely used algorithms for optical flow is based on matching small image patches between frames [Bouguet, 2001]. This approach is not only accurate but fast on a standard processor, and is used throughout the rest of the thesis.

4.3 Motion Modelling

The objective for the rest of the chapter is to develop a method to segment objects based on induced motion. To do this a probabilistic motion model is defined, which captures the correlation between object motion and robotic manipulator. A motion model is calculated for each region of the video, using optical flow, as described in the previous section.

In general terms, let us define a probabilistic motion model as a 3-tuple,

$$\Phi = (Q, \varrho, p(\cdot|Q, \Omega)), \tag{4.4}$$

Where Q is a motion transformation function, ϱ is an error measurement function and $p(\cdot|Q, \Omega)$ is the likelihood of error given a transformation function and some prior knowledge. The motion transformation function, Q^i_j, is calculated as a function of the optic flow of each pixel with the jth segment in the ith frame. For example, Q could be simply calculated as the average optic flow within a segment (as in Section 4.4.1) or as a Homography transformation (as in Section 4.4.1). Then, the position for any arbitrary point in the image, $\boldsymbol{a}$, undergoing this transformation can be calculated as $Q^i_j(\boldsymbol{a})$. The same transformation can be applied to the initial arm position $\boldsymbol{x}_i$, resulting in the expected arm position if this was moving like segment j, $Q^i_j(\boldsymbol{x}_i)$. The transformed arm position can then be compared to the real arm position at frame $i + 1$, $\boldsymbol{x}_{i+1}$.

The error between these two indicates how correlated they are. This error could be defined, for example, by the distance $\varrho(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}; Q) = |Q(\boldsymbol{x}_i) - \boldsymbol{x}_{i+1}|^2$. The probability distribution of the error, $p(\varepsilon|Q, \Omega)$, is then used to assess the membership between the robot and the segment's motion as detailed in the next section. Figure 4-4 illustrates an example where motion transformation functions are defined as the average segment motion.

4.3.1 Assessing Segment Membership through Motion

This section details how the motion of a segment is correlated to that of the robot arm using the probabilistic motion model. The model estimates the probability of observing a particular error magnitude between the end-effector and the segment's estimated motion model. The probability that a particular motion model, Q, explains the observed error is needed, and so an appeal is made to Bayes' theorem to obtain

$$p(Q|\varepsilon, \Omega) \propto p(\varepsilon|Q, \Omega)p(Q|\Omega). \tag{4.5}$$

The posterior conveniently factorises into the likelihood of the error and a prior on the transformation function. The prior is used when extra information is available about the object regarding its shape, colour or location. For example, this could allow the use of previously learned object models to influence the decision about which segments belong to the object being moved. If no prior is known, then Ω is regarded as the empty set.

4.4 Implementation

The method described in this section allows a robotic arm to poke and grasp objects in order to discriminate them from their background. Many of the details regarding robot control and trajectory generation are ignored, as these are not relevant to the contribution. This section details the steps required to practically implement the method.

The motion of regions in the video are extracted and correlated with the arm movement as explained in Section 4.3.1. This gives a probability that each segment belongs to the object, which is then thresholded to extract an object-level segmentation. A prior can optionally be used to give extra information about the object, for instance its colour distribution, or shape.

4.4.1 Motion Transformation Function

The theory laid out in Section 4.3 gives an abstract way to define probabilistic motion models. In this section the definition is applied to the problem of object-level segmentation using a robot arm.

A video stream is first segmented using the low level segmentation algorithm described in Section 4.2.1. The algorithm tracks each segment S_j throughout the video stream. A motion model is then assigned to each segment, j, so that the probability of a segment correlating with the arm movement can be found.

Translational transformation function

Assuming that segments can only undergo a translation motion in the camera space, the model, $\Phi_j^i = (Q_j^i, \varrho, p(\cdot|Q_j^i, \Omega_i))$, can be instantiated as follows:

$$Q_j^i(\boldsymbol{a}) = \boldsymbol{a} + \boldsymbol{b}_j^i \tag{4.6}$$

$$\varrho(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}; Q_j^i) = |Q_j^i(\boldsymbol{x}_i) - \boldsymbol{x}_{i+1}|^2 \tag{4.7}$$

Where Q_j^i is given by a single translation, i is the frame index, $\boldsymbol{a}$ is a variable position vector, and $\boldsymbol{b}_j^i$ the average segment motion. This average motion is calculated using dense optical flow [Bouguet, 2001]. As previously, $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ are the position of the arm at frames i and $i+1$ respectively. The error distribution $p(\varepsilon|Q_S^i, \Omega_i)$ is calculated as a Gaussian with a mean of $\boldsymbol{x}_i - \boldsymbol{x}_{i+1}$ and covariance Σ. The norm given above is the 'L2 norm' $|\boldsymbol{x}| = \sqrt{\sum_{k=0}^{K} x_k^2}$.

Homography transformation function

It is possible to extend the practical implementation into three dimensions. Allowing the robot to segment objects which move non-linearly.

The tensor which models rigid body motion in 3D from observations in a single camera is known as the fundamental matrix [Hartley and Zisserman, 2000]. Points which match across frames $\boldsymbol{x}_i, \boldsymbol{x}_{i+1} \in \mathbb{RP}^2$ are related as follows:

$$\boldsymbol{x}_i^T F \boldsymbol{x}_{i+1} = 0 \tag{4.8}$$

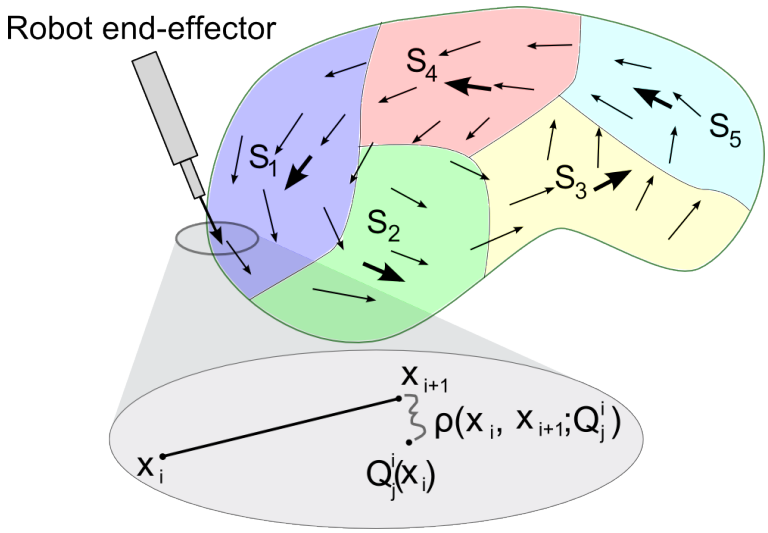


Figure 4-4: A diagram showing how to assess arm motion membership to a model. The arrows in the object denote the optical flow field. Each S_j represents a different segment, each with its own motion model (dark arrow). The large ellipse represents a magnification of a motion model being tested against the motion of the end-effector.

This relation can be used as a motion model in the formulation described in 4.3. Setting $\varrho(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}) = \boldsymbol{x}_i^T F \boldsymbol{x}_{i+1}$, the density function $p(\varepsilon|Q, \Omega)$ can then be accurately approximated as χ^2. The transformation function Q can't be directly computed using this approach, but the implementation only requires the error density function to compute the object segmentation.

If an extra assumption is made that the points inside a segment are planar, then a homography $H \in \mathbb{R}^{3\times3}$ can be used for the motion model:

$$H_j^i \boldsymbol{x}_i = \boldsymbol{x}_{i+1} \tag{4.9}$$

$$Q_j^i(\boldsymbol{a}) = H_j^i \boldsymbol{a} \tag{4.10}$$

$$\varrho(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}; Q_j^i) = |Q_j^i(\boldsymbol{x}_i) - \boldsymbol{x}_{i+1}|^2 \tag{4.11}$$

Both equations 4.8 and 4.10 represent a body moving rigidly in 3D. These models can be used to extract a full 3D model of the object using a single camera. Alternatively, if dense 3D optical flow can be obtained from a pair of cameras then a rigid body transformation function can be used for $Q : \mathbb{R}^3 \to \mathbb{R}^3$.

The probabilities and low-level segmentation algorithm remains the same.

The fundamental matrix can be computed from a pair of homographies [Pellejero et al., 2004]. Combining this approach with the object segmentation algorithm detailed here, and dense monocular SLAM [Newcombe and Davison, 2010], the motion models could be used to compute a reconstruction of the object from the video. This is out of scope for the thesis, but would be an interesting extension for future work.

4.4.2 Error Probability Density Function

Given the model parameters, the error probability density function can be calculated as follows:

$$p(\varepsilon|Q_j^i, \Omega_i) = N(\boldsymbol{x}_i + \boldsymbol{b}_j^i - \boldsymbol{x}_{i+1}|0, \Sigma) \tag{4.12}$$

$$= N(\boldsymbol{x}_i - \boldsymbol{x}_{i+1}|\boldsymbol{b}_j^i, \Sigma) \tag{4.13}$$

$$p(\varepsilon|Q_j^i, \Omega_i) = N(H_j^i\boldsymbol{x}_i - \boldsymbol{x}_{i+1}|0, \Sigma) \tag{4.14}$$

Where Equations 4.13 and 4.14 represent the error probability density function for the translation and homography transformation functions respectively.

The covariance $\Sigma = \lambda I$ is chosen with parameter $\lambda \in \mathbb{R}$. In practise, choosing any value for λ between 10 and 50 does not affect the results significantly.

This definition of the error density assigns a high probability to segments that belong to the robot arm. In order to separate arm segments from those of the object, the next section defines a new probability density function that eliminates highly probable arm segments.

4.4.3 Robust Model Estimation

Estimating dense optical flow is a difficult problem, as noise in the image leads to outliers. Assuming that there are outliers in the data, measures can be taken to improve the quality of the motion model estimation, and is known as *robust* statistics [Tyler, 2008]. There are several methods available to reduce the affect

of outliers, but this chapter is only concerned with calculating the mean optical flow, or the fundamental / homography matrix.

Robust Mean Estimation

For the linear motion model the only parameter needed is the mean, the standard way to compute this statistic for a set of samples $\boldsymbol{x}_i \in \mathbb{R}^N$ is as follows,

$$\mu = \frac{1}{K} \sum_{i=0}^{K} \boldsymbol{x}_i. \tag{4.15}$$

If any number of the $\boldsymbol{x}_i$ are too large then it will significantly affect the estimate of the mean. In order to make this robust to outliers an M-estimator can be used instead,

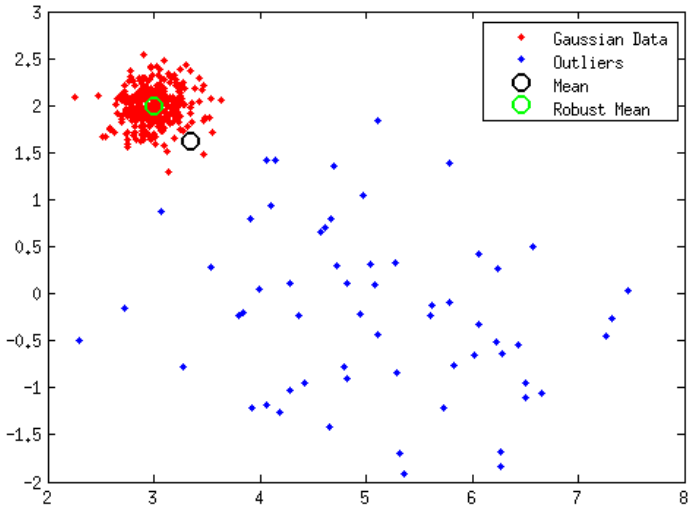
$$\Psi(\boldsymbol{x}) = \begin{cases} \boldsymbol{x} & : |\boldsymbol{x}| < c \\ 0 & : otherwise \end{cases} \tag{4.16}$$

Where c is a constant defining the radius of the M-estimator, and is known as the *bandwidth*. A larger value of c results in more data being taken in to account when estimating the mean, so the convergence may be slow and inaccurate; alternatively, a smaller choice of c may result in an incorrect convergence. Methods for choosing the bandwidth are described in more detail in [Comaniciu et al., 2001]. If the samples are Gaussian distributed with a zero mean, then the robust mean is estimated as,

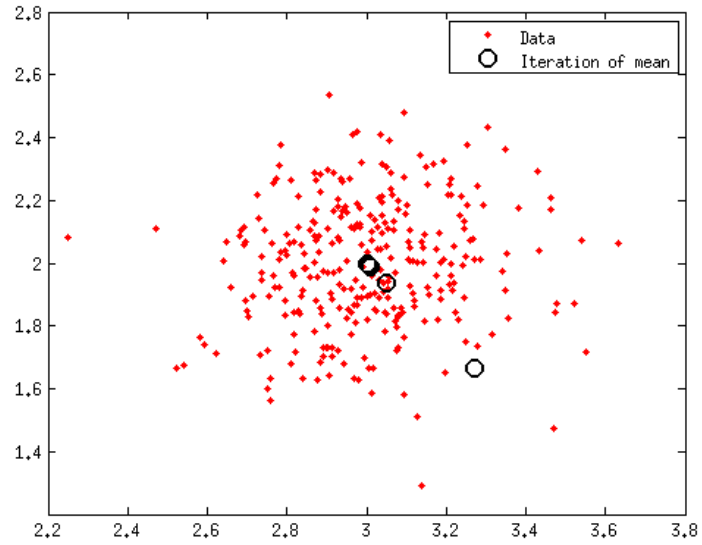
$$\mu = \frac{1}{K} \sum_{i=0}^{K} \Psi(\boldsymbol{x}_i). \tag{4.17}$$

For Gaussian distributed data which is not centred at zero an iterative algorithm can be used to remove outliers and compute the correct sample mean.

This algorithm is an adapted version of mean shift for uni-modal data with a flat kernel, proven to iterate to the mode regardless of the distribution [Cheng, 1995] for a well chosen value of c. Figure 4-5 shows an example of this for Gaussian samples corrupted by outliers.



(a) Robust Mean Estimation



(b) Convergence

Figure 4-5: These figures show how the mean can be computed in the presence of outliers. Figure 4-5a contains data which is normally distributed (red) and from another distribution (blue). Iteratively windowing the data, computing the mean and centring the window around the new mean, accurately finds the *correct* mean for the normal data. Figure 4-5b shows that the algorithm converges. The axes do not show the units since the plots are to be analysed relatively.

Algorithm 1 Robust calculation of the mean

Require: c, $\boldsymbol{x}_1$, ..., $\boldsymbol{x}_N$, Q
  $\mu \leftarrow \frac{1}{K}\sum_{i=0}^{K} \boldsymbol{x}_i$
  $\varepsilon \leftarrow \infty$
  while ε >Q **do**
    $\mu_T \leftarrow \mu$
    $\mu \leftarrow \frac{1}{K}\sum_{i=0}^{K} \Psi(\boldsymbol{x}_i - \mu) + \mu$
    $\varepsilon \leftarrow |\mu - \mu_T|$
  end while

Random Sampling Consensus

For more complicated motion models, robust regression is needed instead. A popular algorithm used for this purpose is found in the computer vision literature, and follows a similar procedure to algorithm 1.

Suppose we have a transformation function Q taking points in one frame to points in the next. A random set of correspondences are chosen and the parameters of Q are computed. The metric is then used to determine the error over all of the other correspondences. The transformation function which gives the largest number of inliers is chosen as the correct transformation function. The Random Sampling Consensus (RANSAC) method is detailed in Algorithm 2.

Algorithm 2 RANSAC robust estimation [Hartley and Zisserman, 2000]

Require: $\boldsymbol{x}_1$, ..., $\boldsymbol{x}_N$, $\boldsymbol{y}_1$, ..., $\boldsymbol{y}_N$
  $k \leftarrow 0$
  while k <K **do**
    a) Select a random sample of correspondences and compute the transformation matrix Q
    b) Calculate the distance ϱ for each putative correspondences
    c) Compute the number of inliers consistent with Q by the number of correspondences for which $\varrho < t$ pixels
    $k \leftarrow k + 1$
    Update K
  end while
  Choose Q with the largest number of inliers. In the case of ties choose the solution that has the lowest standard deviation of inliers.

4.4.4 Removing Arm Segments from the Object

Correlating all movement in the video with that of the arm results in the arm itself being segmented as a part of the object of interest. To prevent this from occurring, the arm segments must be eliminated from the video.

The standard way to remove the arm segments would be to use an accurate 3D model of the arm and project it onto the cameras [Welke et al., 2010]. The work presented in this chapter presents a probabilistic approach which is a simpler alternative, as calculating the kinematics and calibrating a full CAD model of the robot is time consuming, and assumes that an accurate model of the robot is available. Further, the method used here could be combined with model-based arm removal for improved accuracy and robustness.

Conceptually, any motion that correlates with the arm movement before an object is touched or moved is likely to be a part of the arm. This information is used to remove the arm from the segmentation. Thus, the full error distribution is:

$$p(\varepsilon|Q_j^i, \Omega_i) = \begin{cases} \frac{1}{\kappa_j^i}(1 - p(\varepsilon|Q_j^i, \Omega_i)) : i \in T' \\ p(\varepsilon|Q_j^i, \Omega_i) : i \notin T' \end{cases} \tag{4.18}$$

with,

$$\kappa_j^i = \int_{\boldsymbol{x}_i, \boldsymbol{x}_{i+1} \in D} 1 - p(\varrho(\boldsymbol{x}_i, \boldsymbol{x}_{i+1}; Q_j^i)|Q_j^i, \Omega_i) \tag{4.19}$$

Where $D = [-N, N] \times [-M, M]$ is the set of all possible values for the optical flow. M and N are the height and width of the image respectively. In practice κ_s^i can be any constant since each i is independent. In this formulation $\Omega_i = (i, T, \Sigma)$. In Figure 4-6 it can be seen that the robot arm segments (left side of the images) are darker (lower probability) in comparison to the other segments in the image.

4.4.5 Integrating over Multiple Frames

So far we have only considered the probability of a segment in a single frame. It remains to calculate the distribution for a set of frames. Using Equation 4.5 and substituting Ω by $\Omega_j^i = (i, T, \Sigma)$ and Q by Q_j^i allows prior knowledge to be

incorporated at each frame i. The goal is to segment the object throughout the whole video, so the a probability of a segment belonging to the object is sought, independently of any particular frame. The variable i can be eliminated by marginalisation over T, using the law of total probability [Bishop, 2006]. This step can be realised at any point in time, for a set $\hat{T} \subset T$:

$$p(S_j|\varepsilon, T', \Sigma) = \sum_{i \in \hat{T}} p(S_j|\varepsilon, i, T, \Sigma) p(i|\varepsilon, T, \Sigma) \tag{4.20}$$

$$= \frac{1}{|\hat{T}|} \sum_{i \in \hat{T}} p(Q_j^i|\varepsilon, \Omega_i) \tag{4.21}$$

The final step follows from assuming that our confidence in the variable i is uniformly distributed over the whole window $\hat{T}$, and that the probability of a segment S_j given frame i is equal to that of its motion model Q_j^i.

This results in a probability map for each segment throughout the video. The quality of the object segmentation improves as this probability is integrated. Figure 4-6 shows the probability map evaluated for a range of different $\hat{T}$. The left most image shows the distribution before the object has been touched, and on the right after the integration over 21 frames. The centre image is the distribution at the time the robot touches the object, marked with a red border.

Thresholding the probability map to obtain a segmentation throws away a lot of information. In practise a map is stored as a measure of our confidence in the whereabouts of the object. The robot could then go back and touch areas of the scene that it is uncertain about, transforming the process into interactive segmentation [Kenney et al., 2010].

4.5 Sparse Features

When programming a robot to learn about objects, descriptive features may be more useful. The method of using dense optical flow, calculated at every pixel, and low-level segmentation gives a very accurate segmentation of the object being manipulated. Although the algorithm accurately finds the boundary of the object, there is a trade off in speed. When learning about an object using

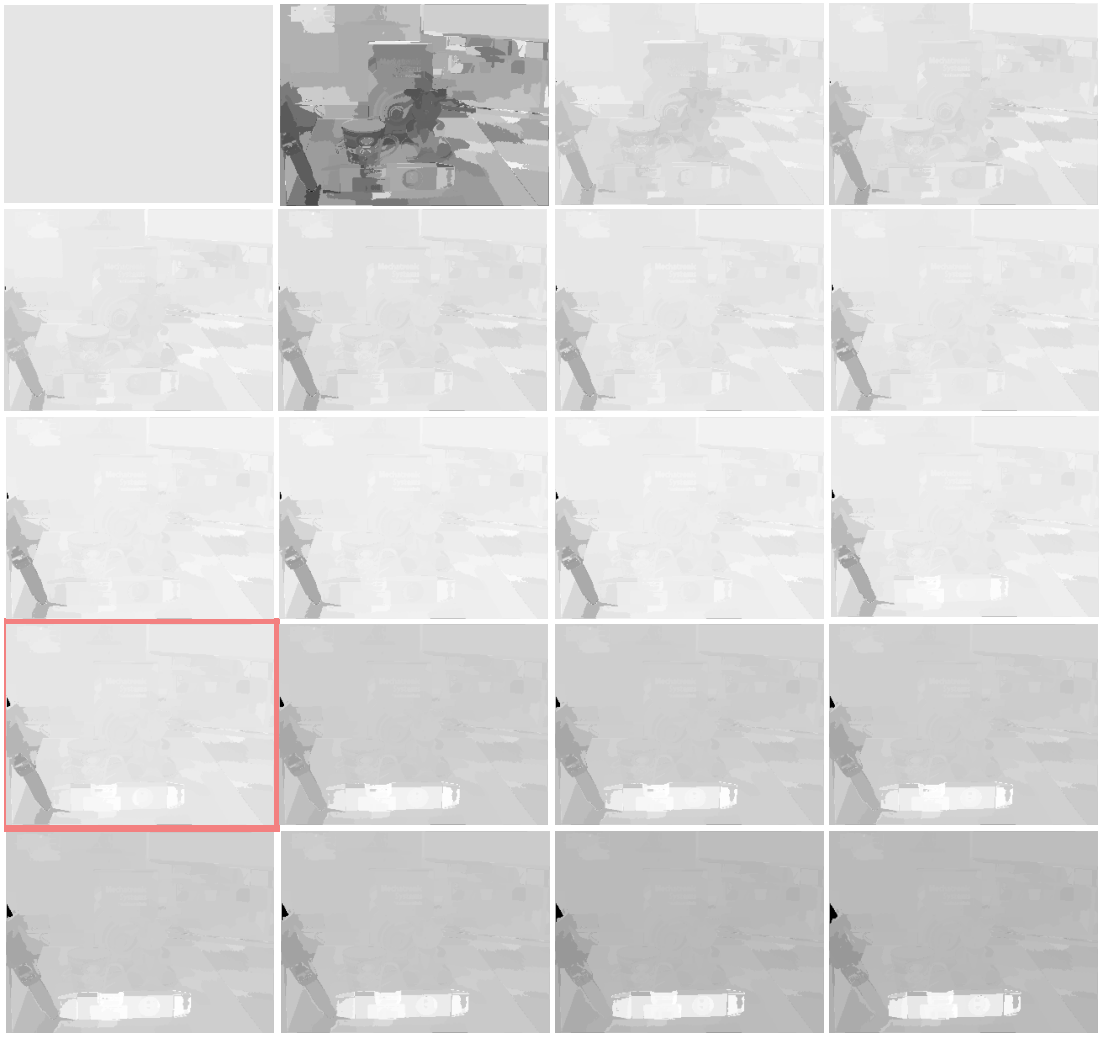


Figure 4-6: The probability map of a spirit level being pushed, evaluated at a different set of times throughout the video. The top-left image is the map for the first frame and the bottom-right represents the final frame. A red boundary shows the first frame the object was touched. The spirit level becomes more probable (brighter) with each successive frame.

feature points, it is not necessary to find the boundary of the object in the camera space, only the feature points that belong to the object.

Descriptive features (such as SIFT [Lowe, 2004b]) find sparse points on an image frame and describe them so that they can be matched to corresponding points in successive frames. There are two points that make this different from the optical flow and low-level segmentation approach defined above,

1. Firstly, optical flow is dense, computed at every pixel in the image. Sparse features can be used and matched across frames as an alternative method to standard optical flow.

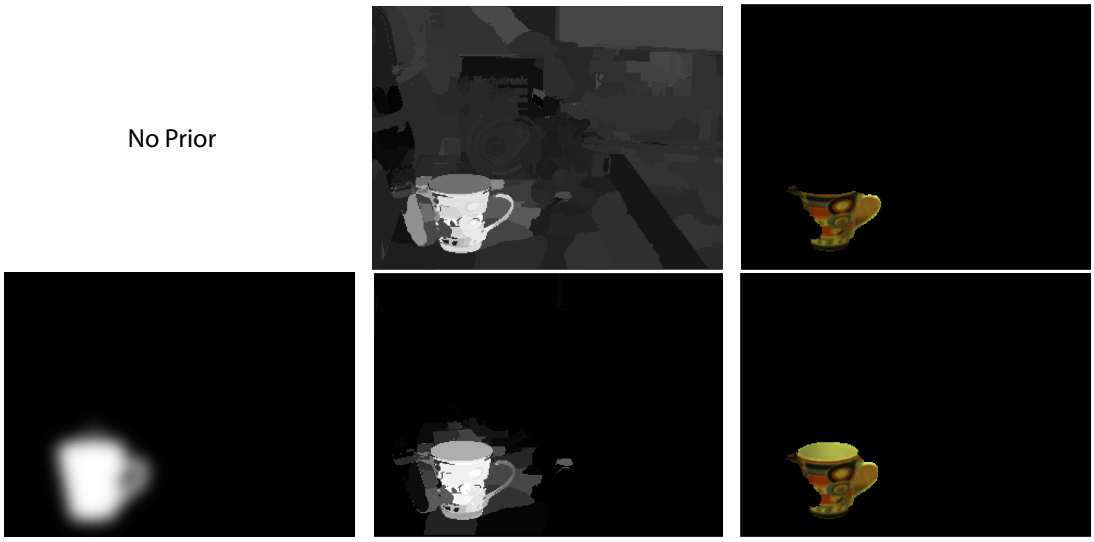
2. Since the features are sparse, there may be large regions of the image that do not contain any features at all. Using this in combination with low-level segmentation is difficult as some segments may not contain any features and so the motion information can't be calculated. Instead, a low-level segmentation is performed on the features themselves, rather than the frames. The features don't need to be tracked throughout the image, but the descriptors output from the frame by frame segmentation can be fed directly into a learning algorithm.

Segmenting sparse features follows the same motion modelling concept introduced in previous sections. The method is outlined in Algorithm 3. Where $\boldsymbol{v}_1$, ..., $\boldsymbol{v}_N$ are the velocity vectors between matching feature points, $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i-1}$ are the arm positions at frame i and $i-1$ respectively. The parameter τ is a threshold to differentiate between features moving with the arm and background.

Algorithm 3 Sparse feature segmentation

Require: $\boldsymbol{v}_1$, ..., $\boldsymbol{v}_N$, $\boldsymbol{x}_i$, $\boldsymbol{x}_{i-1}$, τ
  C=meanShift($\boldsymbol{v}_1$,$\boldsymbol{v}_2$,... ,$\boldsymbol{v}_N$)
  for k=1 $\rightarrow$ size(C,2) **do**
    label(k) $\leftarrow N(C_{k:}|\boldsymbol{x}_i - \boldsymbol{x}_{i-1}, \Sigma)$
  end for
  foreGroundFeatures $\leftarrow$ find(label $\geq \tau$)
  return foreGroundFeatures

The segmentation algorithm works well when every feature is matched to its corresponding feature in the consecutive frame. Noise in the images lead to



(a) Prior (b) Probability Map (c) Segmentation

Figure 4-7: Using a position prior to segment a cup. The top row shows the object segmentation with no prior. The bottom row has a strong position prior in one of the frames, leading to a much better final segmentation.

some features not properly matching their counterparts, however. In order to improve the result, density estimation is used on the spatial position of the features, and then a threshold is applied. This takes all of the features that were moving in the same way as the arm and finds the nearby features, bringing them into the foreground.

4.6 Using Prior Knowledge

Although the algorithm works without any prior knowledge of the objects in the environment. The Bayesian formulation allows priors to be used to reinforce or improve the final segmentation. For instance a basic background or arm model could be used to further discriminate the object of interest from its surroundings.

This allows the use of a weak recognition algorithm to infer the existence of an object and its whereabouts, based on colour or texture information. The robot can then push the object to segment it based on the priors given by the recognition algorithm. Part III of this thesis is dedicated to recognition and

classification, the prior provides a coupling between the bottom up segmentation and top down recognition.

4.7 Theoretical Discussion

The method and theory laid out in this chapter introduces object level segmentation using knowledge of the robotic arm motion, a sense of proprioception. The extra information allows the robot to both localise and segment the object it is interacting with, removing background from the video. A general segmentation algorithm can separate a video into composing objects, but is not able to determine which object is being manipulated by the robot.

In section 4.4.1 the complexity of the algorithm is mentioned. There are K frames and L unique segments in the video, and the mean flow vector $\mathbf{b}_S^i$ needs to be calculated for each, giving KL motion models. Unless the distribution is updated for each frame as the video stream is being processed, the complexity will rise linearly with the number of frames. Using this method for large sections of video could become computationally difficult. To ensure the algorithm is practical for robotic applications it can be modified to work incrementally. Section 4.4.5 shows how the distributions in multiple frames are integrated into one. This process is equivalent to building the distributions at each frame using a running average.

It may also be the case that we have some measure which gives a confidence in particular frames of the video, the robot may have some prior knowledge about any noise in the images which affect its vision negatively. This can be formulated as a prior probability on the frames, and incorporated into Equation 4.21, further increasing the stability and robustness of the final object segmentation. Again, this is not considered here, but using prior knowledge about the robots sensors and objects in the scene is something that will be incorporated into future work.

Chapter 5

Experiments in Object Segmentation

The theory introduced in the previous chapter leverages the known motion of a robotic arm in order to allow a robot to both localise and segment objects. The algorithm allows a prior object model to be used, in order to improve the segmentation results, and will theoretically remain robust to cluttered and moving backgrounds.

Current object segmentation algorithms which are based on robotic motion assume that the background is static. The method improves current methods by improving quality of segmentation when the background is moving. The method considered here is tested against a current state of the art algorithm in the area, demonstrating the improvement both qualitatively and quantitatively.

To summarise the experiments carried out in the chapter. The method is tested against the current state of the art algorithm for robotic object segmentation in both,

- **Static backgrounds**: The algorithm is robust to videos which are cluttered but have static backgrounds. The results are comparable to state of the art.
- **Moving Backgrounds**: The algorithm performs significantly better than previous results when the background is moving.

Alongside the experiments mentioned above, the method is also tested for objects undergoing significant rotation, and also, when the object is

manipulated in three dimensions. The work concludes in a tested algorithm which is able to segment objects in the presence of clutter and other moving objects in the environment surrounding the robot.

5.1 Translation Models

The method is tested using a single camera and a Katana 5 DOF robot. A total of 61 videos of the robot pushing 5 different objects were taken (toy rabbit, mug, spirit level, bottle and book). In this set of videos the robot moves linearly and parallel to the table that contain the objects (see Figure 3-1 for the experimental set up). The videos are of a cluttered scene, some with moving backgrounds. The translation transformation function defined in Section 4.4.1 is used here.

At each frame, the position of the arm's end-effector is recorded together with a binary variable indicating the frames in which the object is being pushed. The frame in which the objects are touched for the first time are hand segmented to give a ground truth segmentation to evaluate the results. This dataset can be obtained by emailing the authors of this thesis.

5.1.1 Comparing to current state of the art

The method is compared experimentally to [Fitzpatrick and Metta, 2003] since they use robot motion to segment objects. Although [Kenney et al., 2010] provide a more recent version, the underlying segmentation algorithm is the equivalent to that presented in [Fitzpatrick and Metta, 2003]. In both approaches, subtraction is used to determine the temporal location of the touch event. The frame directly preceding the touch is used as a model for the arm. The Graph Cut algorithm (GraphCuts) [Boykov and Kolmogorov, 2004] is then used to build a hull around these points. The frame in which the object is touched for the first time is applied to GraphCuts, with the pixels that belong to the arm model classified as background.

Some results of our algorithm, named Motion-Likelihood (M-L), are shown in Figure 4-1, showing the full process used to obtain the object-level segmentation. The far left column shows the initial videos of the toy rabbit, cup

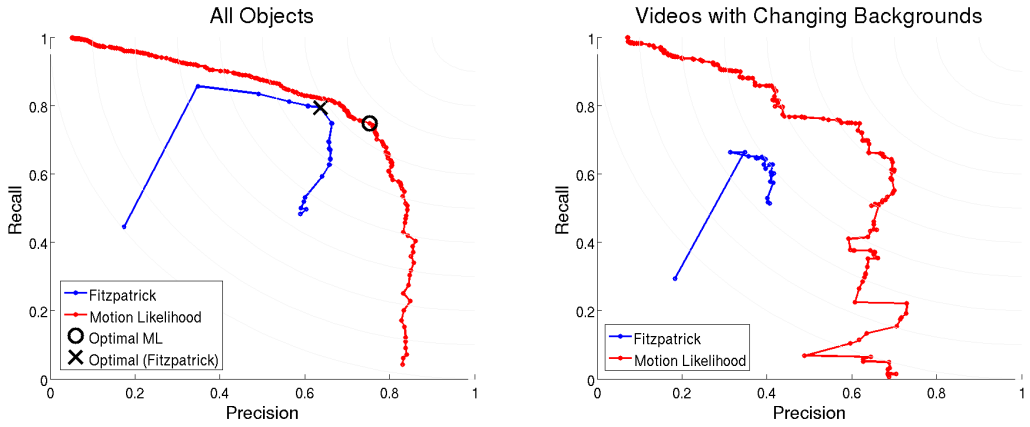
and spirit level which are being poked by the robot; a low-level segmentation is obtained using the algorithm of [Paris and Durand, 2007], shown in the second column; the probability map is given in column three; and the result of thresholding the map is displayed on the far right. It is worth noting here that [Fitzpatrick and Metta, 2003]'s method can only be computed on the frames nearest to the touch event, whereas M-L gives a segmentation for the object throughout the whole video, this means that M-L has the advantage that more information can be used to discriminate between background and object. Throughout the results, the same frame segmented using [Fitzpatrick and Metta, 2003] is used for comparison against M-L.

Both M-L and [Fitzpatrick and Metta, 2003]'s algorithm rely on a free parameter. In [Fitzpatrick and Metta, 2003], the authors apply a threshold to the subtracted image. M-L applies a threshold to the probability map. Reasons for choosing this parameter may vary according to the hardware used or visual statistics of the scene. Following the work of [Martin et al., 2004], the segmentation's *Precision* and *Recall* are calculated for every value of the free parameter against the ground truth. These graphs will indicate the optimal parameter values for both segmentation algorithms and compare their performance. Precision (Pr) and Recall (Re) are defined as follows:

$$Pr = \frac{t_p}{t_p + f_p} \tag{5.1}$$

$$Re = \frac{t_p}{t_p + f_n} \tag{5.2}$$

Where t_p is the number of true-positives, the pixels that are correct and verified by the ground truth; f_p is the number of false-positives, pixels which are not part of the ground truth but have been detected as being part of the object; and f_n are the number of false-negatives, pixels which are in the ground truth but are not found to be part of the object. If the true object is completely covered by the segmentation then there will be no false-negatives and the recall will be 1, on the other hand, if the object covers the segmentation then there will be no false-positives and the precision will be 1. The segmentation is perfect if and only if both Precision and Recall have a value of 1.



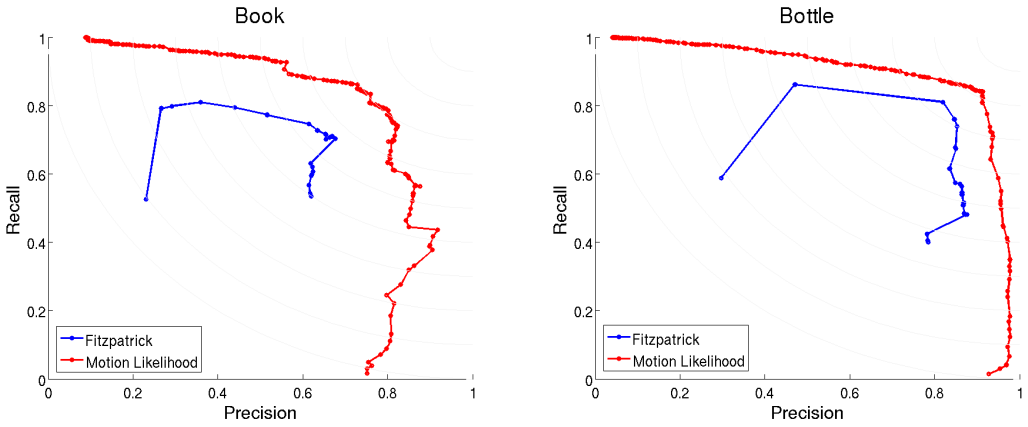
(a) All Objects (b) Changing Backgrounds

Figure 5-1: P-R graphs for M-L and [Fitzpatrick and Metta, 2003]. With static (a) and changing (b) backgrounds.

The results of using Precision-Recall (P-R) for every parameter of M-L and the algorithm from [Fitzpatrick and Metta, 2003] are shown in Figure 5-1 for the five different objects. Figure 5-1(a) is the P-R for videos with static background and Figure 5-1(b) with a changing background. The points on the line represent different values of the parameter. The segmentation is of a better quality if it is closer to the top right hand side of the graph (1,1). Both of the graphs show that M-L has a superior segmentation performance to that of [Fitzpatrick and Metta, 2003]. Figure 5-1(b) shows the P-R graph for all of the videos with a changing background. In this test M-L performs significantly better.

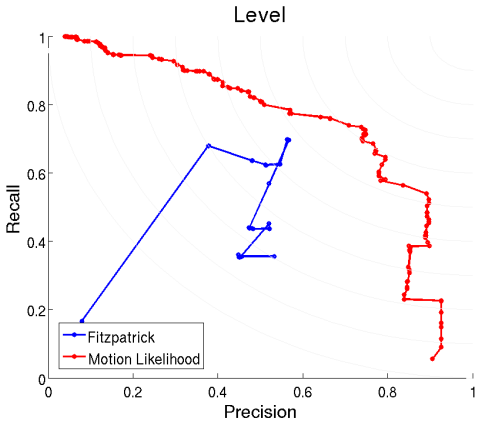
The P-R graphs shown in Figure 5-2 show segmentations of each object for both moving and static backgrounds. It is clear that M-L performs as well as [Fitzpatrick and Metta, 2003] for all objects other than the Cup. This is likely to be due to the highly patterned surface, causing segmented regions on the mug to be small, thus, increasing the noise in calculating the motion models. The Book, Bottle and Spirit Level perform noticeably better, and the Rabbit is of no significant difference.

Low valued thresholds result in most of the background being considered as part of the object, thus has a high recall. M-L exhibits this relationship, with recall decreasing as the threshold increases. The P-R results for [Fitzpatrick and Metta, 2003] show a graph with a more unusual shape at a low threshold.

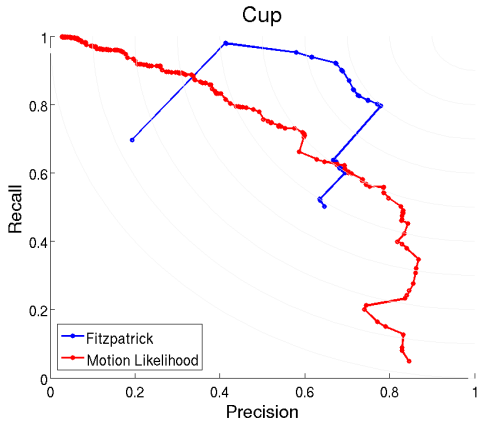


(a) Book

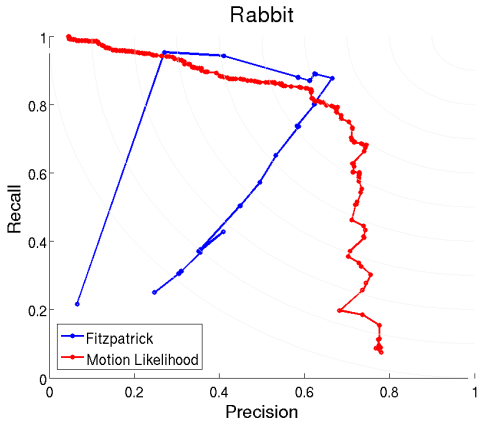
(b) Bottle



(c) Level



(d) Cup



(e) Rabbit

Figure 5-2: P-R graphs for M-L and [Fitzpatrick and Metta, 2003]. For each of the objects used for testing. M-L works very well for all objects except the cup, which is covered by a highly textured pattern.

This was further investigated and it was found that when the subtraction method used by [Fitzpatrick and Metta, 2003] is thresholded at a low value the result is a noisy binary image with foreground distributed across the whole image. Proceeding to use GraphCuts on the image acts as a filter removing a lot of the background noise but inaccurately segmenting the object. As the threshold is increased, the amount of noise reduces, significantly improving the segmentation and the result from GraphCuts.

Although testing the algorithm across each parameter gives good intuition of the performance, a single parameter for the threshold needs to be calculated to verify the performance for practical use.

To determine the quality of the segmentation at a single threshold we introduce a measure of the distance between foreground/background segmentations and ground truth, formally the average L2 distance between two binary images:

$$\rho_{L2}(Im_1, Im_2) = \frac{1}{NK}\sqrt{\sum_{i=1}^{N}\sum_{j=1}^{K}|Im_1(i,j) - Im_2(i,j)|^2} \tag{5.3}$$

and the Tanimoto, or Jaccard, distance [Clarkson, 2006], given by,

$$\rho_T(Im_1, Im_2) = 1 - \frac{\sum_{i=1}^{N}\sum_{j=1}^{K}|Im_1(i,j) \wedge Im_2(i,j)|}{\sum_{i=1}^{N}\sum_{j=1}^{K}|Im_1(i,j) \vee Im_2(i,j)|}. \tag{5.4}$$

The threshold for each algorithm is chosen to be the optimal value in terms of the Precision and Recall. For both M-L and [Fitzpatrick and Metta, 2003] the point on the P-R curve for all of the objects that is closest to the point $(1, 1)$ is chosen. This optimal value is shown in Figure 5-1(a), and takes on a value of 75% of the maximum likelihood for M-L and a threshold of 5 for [Fitzpatrick and Metta, 2003]. These values are used to segment each video in the dataset for comparison. The measure in Equation 5.3 is used to find the distance between the object segmentations and ground truth, giving the average distance between two segmentations.

This is done for all of the videos using [Fitzpatrick and Metta, 2003]'s method, M-L and M-L with the Graph Cut algorithm (ML+GraphCuts). Using GraphCuts draws a hull around the segmented object in order to fill in any

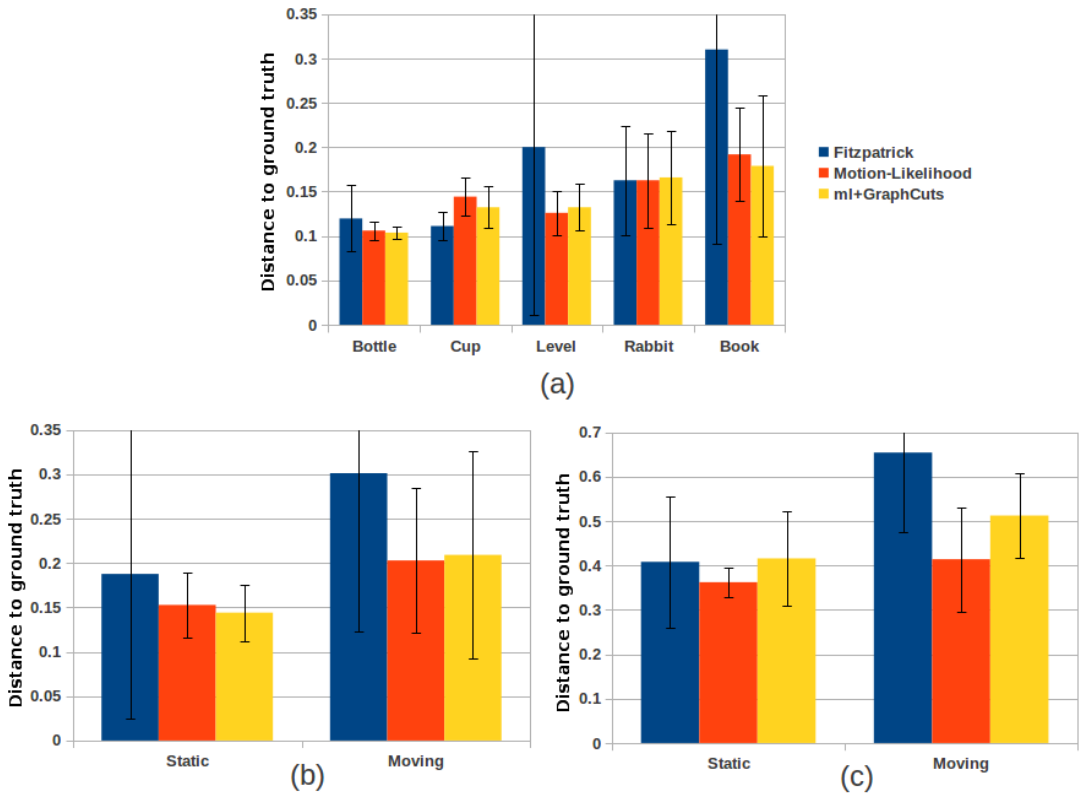


Figure 5-3: Bar chart comparing segmentation algorithms for both changing and static backgrounds. Each colour represents a different method; the y-axes represent the distance from the computed segmentation to the ground truth, with error bars representing the standard deviation. Organised by different objects (a) and backgrounds (b), measured by the L2 distance defined in Equation 5.3. The bar chart in (c) represents the errors as measured by the Tanimoto distance in Equation 5.4

details which are missing in the original segmentation. Experimenting with this algorithm determines how well the M-L algorithm performs, if GraphCuts makes a big difference to the result then the M-L performance is poor for details on the object.

Figure 5-3 shows the results of this experiment. In Figure 5-3(a), for the book, spirit level and bottle, M-L provides a better quality of segmentation. The low standard deviation implies it is more stable across a range of different environments. Videos in the dataset include moving backgrounds, camera shake and change in lighting condition. The rabbit performed roughly the same, and the cup object performs slightly worse because of a failure in the low-level

segmentation algorithm of [Paris and Durand, 2007], due to the complicated pattern on it.

The results were further divided into cases in which the video has a static background and ones with a changing background. Figure 5-3(b) shows that both methods become a little more unreliable when the background is changing, M-L represents a significant improvement over [Fitzpatrick and Metta, 2003] however.

Figure 5-3 visually indicate that our method offers an improvement over [Fitzpatrick and Metta, 2003], the Kolmogorov-Smirnov (KS) test [Massey Jr, 1951] provided a quantitative indicator. In the context of segmentation, this can be interpreted as the percentage chance that our method outperforms [Fitzpatrick and Metta, 2003]'s, given a random scene. In the case of static backgrounds the KS-test gives results of 60% and 45% for ML and graph cuts. For moving backgrounds we get 48% and 95%. The L2 distance, is known to be a poor measure of visual quality; the Tanimoto distance, as defined in Equation 5.4, is thought to be more reliable for binary images. Figure 5-3(c) graphically displays the results. The KS results for the Tanimoto distance are as follows, 46% and 61% for ML and graph cuts in static scenes, and 81% for both ML and graph cuts for moving backgrounds. Presenting qualitative results help one to interpret these numbers.

Figure 5-4 shows a sample of the results for videos with a static Figure 5-4(a) and changing Figure 5-4(b) backgrounds. Both methods perform reasonably well when the background is static, although using subtraction poses problems in some of the videos. For example, in the centre column of Figure 5-4(a). The constant uniform yellow colour of the spirit level means that image differencing does not detect a change when the object is moved between frames. The result from GraphCuts is shown in this image, [Fitzpatrick and Metta, 2003] goes on to extract the connected region which is closest to the end-effector when the object is touched for the first time, further degrading the quality of the final segmentation. Problems like these contribute to the wide standard deviation shown in Figure 5-3(b).

The changing backgrounds have a significant effect on the subtraction used by [Fitzpatrick and Metta, 2003] (bottom row), and in some cases results in the whole image being segmented as part of the object as in Figure 5-4(b). Our

(a)

(b)

Figure 5-4: Different segmentations tested on static (a) and changing (b) background. The top row are the results for M-L, the centre shows the probability map the bottom row are results of the method described by [Fitzpatrick and Metta, 2003].

results show a large improvement (top row). [Fitzpatrick and Metta, 2003] failed for a number of different reasons, the right hand column of Figure 5-4(b) shows a frame from video with camera shake. In the fourth column, movement of the book causes the object behind to move slightly. Columns two and three include people moving behind the robot. Any level of movement or change in the environment causes the background subtraction used in [Fitzpatrick and Metta, 2003] to fail.

5.2 Homography Models

Extra datasets are used to test the method on objects undergoing challenging motions. 10 videos of the robot pushing a bottle on a table and undergoing significant rotations are used to assess the performance of the algorithm under non-linear motions. Two further datasets (10 videos each) containing a box and spirit level being gripped and moved in space by the robot are used to test the model under 3D motions. In these experiments, both the translation and homography transformation methods are compared. The results are compared to a hand segmented ground truth for a random frame in the video sequence.

The qualitative performance of the algorithm using a homography motion model is shown in Figure 5-5. The columns show a frame of the video, the probability map and resulting segmentation respectively. The first row shows a segmented frame of the video where the object undergoes significant rotation during the push. The second and third rows are videos of an object (spirit level and box) being held by the robot and moved non-linearly in front of the camera.

Figure 5-6 shows the Precision-Recall graphs for a homography model and the translation model. The graphs show a similar performance for both models for the pushed bottle Figure 5-6(a). This is due to (i) inaccuracies in calculating the homography for small movements in a segment, and (ii) the good approximation of translations only for small motions.

Figures 5-6(b) and 5-6(c) show results for objects which are held by the robot and moved non-linearly and in 3D in front of the cameras. Here, the homography based method shows a significant improvement over the translation model.

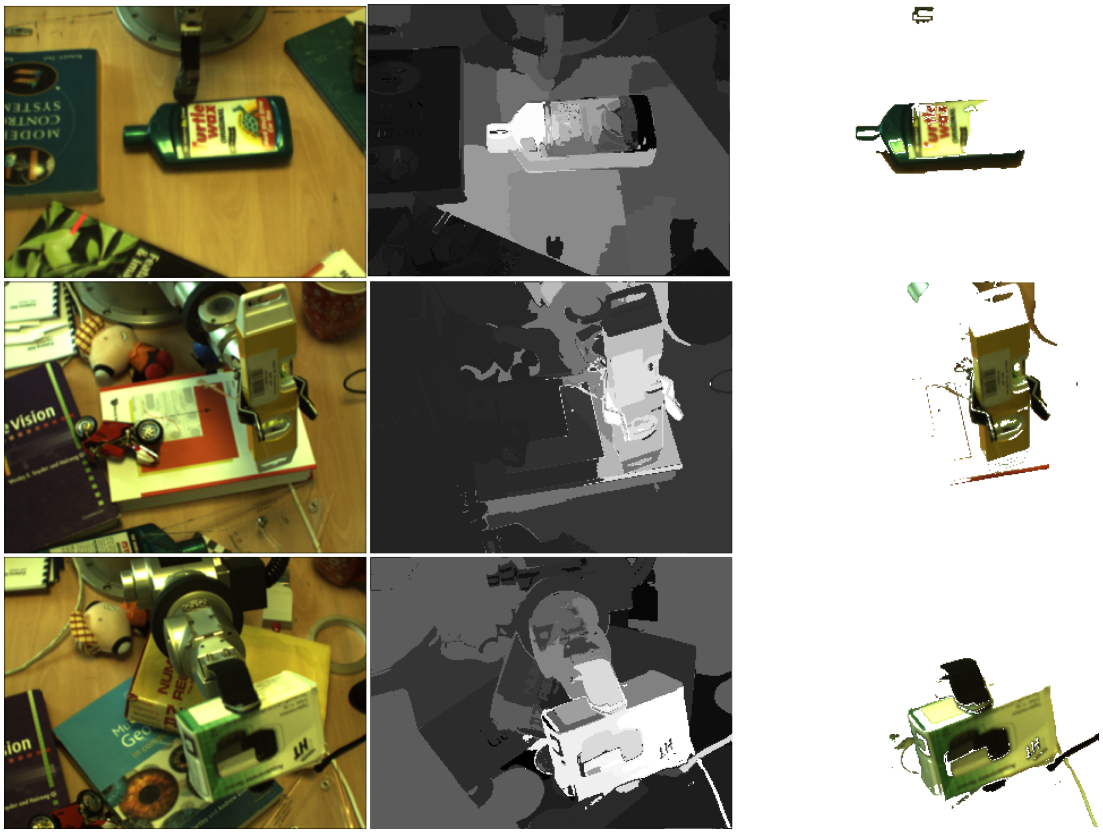
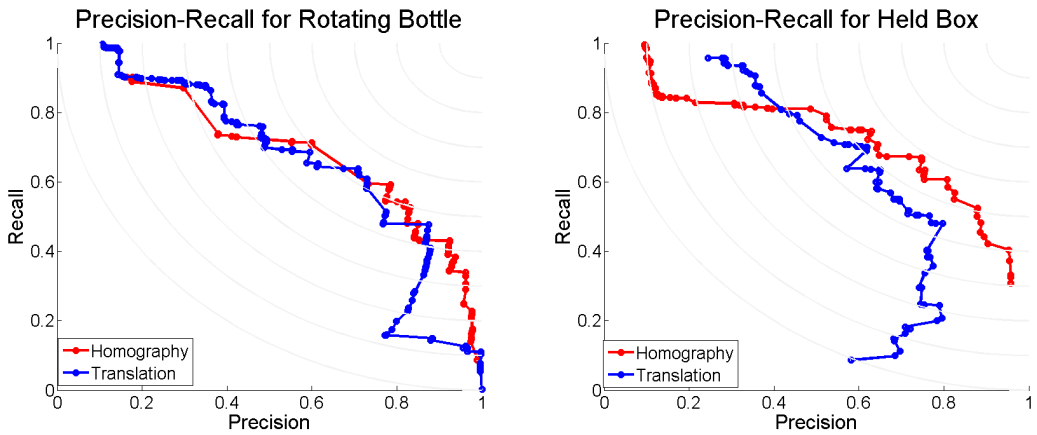


Figure 5-5: Segmentations and probability maps computed using a homography motion model.

The spirit level tested is accurate near the manipulator, but becomes less accurate towards the edges. This is due to the quality in estimating the homography further away from the manipulator. High velocity motion at the edge of the level is approximately linear in these regions and, if it is calculated inaccurately, will not match well with the arm motion. In Figure 5-6(b) a smaller object is used, with improved precision and recall. The linear model also works much better in this case, but is still out performed by a homography.

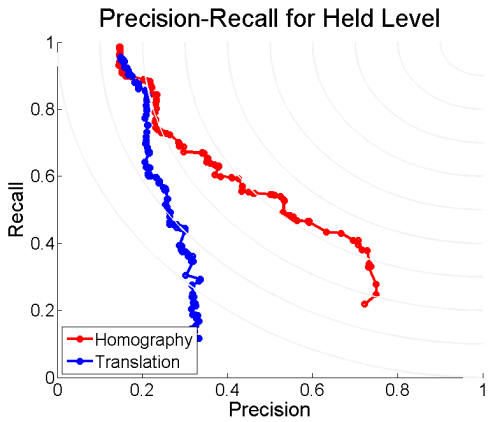
5.3 Discussion

A new method for segmenting objects based on visual information and robot motion has been introduced. Alternative state of the art methods use image differencing as a basis for the segmentation and despite its ease of use and



(a) Rotating Bottle

(b) Held Box



(c) Held Level

Figure 5-6: P-R graphs comparing Homography and Translation motion functions. (a) Rotating bottle, (b) held box and (c) held spirit level. The results are comparable, except in the case of the spirit level, for which a homography motion model works much better.

ability to cope with cluttered, or unstructured backgrounds, break down if the background environment is dynamic. Empirical results show that the method provides a much better quality of segmentation compared to previous results, with both, dynamic and static backgrounds. To the best of our knowledge, this is the first implementation of motion-based segmentation using a robot where the background can change and no prior knowledge is used.

The results also illustrate how the method can be extended from poking objects in a linear fashion to cases in which the objects are moved in complex

3D trajectories. This is based on a homography motion model which assumes that points within segments lie in a plane. This assumption could be relaxed if full 3D data (i.e. stereo camera system) was used. In this case, a standard 3D rotation-translation matrix could be used as a motion model for rigid objects.

Flexible and articulated objects would require complex non-linear motion models for their segmentation. In principle, the method introduced here could be recursively used for each connected component of an articulated body. Although beyond the scope of this thesis, this is an interesting area for future research.

The probabilistic formulation of the algorithm gives a measure of certainty in regions of the image belonging to the object. This will form a basis for the robot to decide the best place to manipulate the object from in future, further differentiating the object from it's background.

In summary, the contribution made in this chapter to the current literature is as follows:

- Support for the hypothesis that robotic motion can be used to localise and segment objects manipulated by the robot.
- The background is not assumed to be static, allowing the method to perform better in changing environments than previous methods.
- The segmentation is probabilistic, giving a degree of certainty in any part of an image being part of the object.

The segmentation algorithm provides a way to filter out background noise in a changing environment. Using it on a robotic system in this way provides the first step towards a machine that can start learning about objects. The next part of the thesis explores techniques in machine learning for describing the visual objects in the robot's workspace.

Part III

Robot Learning

Chapter 6

Autonomous Visual Learning of Object Models

In the previous part of the thesis, a method for extracting an object from the background was presented. This allows the robot to differentiate the object it is playing with from other distracting objects, as well as the robot itself. For a robot to classify the objects, it needs to turn the pixel data into a representation which can be used to recognise the object again from a variety of poses, under different illumination and also in the presence of occlusions. This part of the thesis deals with the building of such models, using a robotic system.

The part is broken into three chapters, representing the three main contributions listed below.

- **An Online version of a successful visual classifier is adapted for a robotic vision system**. The classifier used here is known as Latent Dirichlet Allocation (LDA), and has been recently modified to learn Online [Hoffman et al., 2010b]. The use of a *universal dictionary* is explored, and results are compared against standard LDA, and also using a robot vision system.

- The robot can then use decisive planning and motion to manipulate the objects in its workspace, moving objects out of the field of view, and rotating objects of interest. This process is termed **self-supervision**, and allows the robot to use a supervised classifier for unsupervised learning. Since the robot can supervise its own learning, a new **Mixture of Topics**

model is introduced, an extension to LDA showing improved classification results.

These two contributions are covered in Chapters 6 and 7. This chapter introduces visual object learning using Online LDA, and provides the basis for a robot which can learn autonomously. It is not in direct support of the hypothesis, but contributes a novel visual learning algorithm for robotics. The second contribution, covered in Chapter 7, is in support of the hypothesis since it shows that manipulation of objects can be used to self-supervise. This leads to better classification results and object models built from a number of different views.

The final chapter in this part,

- Combines the **segmentation** algorithm built in Part II with the **Mixture of Topics** model built in Chapter 7. This inherits the qualities of both algorithms so that the robot can autonomously learn robust visual object models, in the presence of moving backgrounds.

Visual learning is a difficult, and largely unsolved, problem. The process involves extracting defining information about objects from the image, and representing it as an object model in a compact manner. A robot will not necessarily have all the data needed to build visual models of its environment. If it were to use Batch learning then it would need to save data for everything it has already seen, and recompute for each new set of data. This is remedied by the use of Online learning, in which the object models are updated as new data becomes available.

This chapter begins by introducing 'bag-of-words' models for visual learning, the paradigm on which LDA is based, and details the approach used in this thesis. The problems associated in using the approach for robotic vision are covered. Finally, the approach is tested on the Caltech dataset, and for objects placed in the robot's workspace.

6.1 Bags of Words for Visual Learning

There is a well known set of methods for classification in the text literature, known as 'bag-of-words' classifiers. In these approaches, a dictionary is formed

from all of the documents, and a document is represented as a histogram of its words over the dictionary. This approach is explained with mathematical rigour in Appendix B.1.

Although the statistical model for learning is taken from the literature on text analysis, it can be applied to computer vision. The problem is slightly different, however, since images are not made up of discrete words, but instead, an array of coloured pixels. In order to use bag-of-words methods for vision problems, the data needs to be processed so that it fits into the same framework. Essentially, this means processing the image so that it can be represented as a histogram.

To summarise, the rest of the section is broken into the following subsections

- **Features**. The visual features represent the meaningful information in an image. The goal is to remove as much of the information that is variant to change in viewpoint, illumination and occlusion as possible, leaving only information that can be used to represent the objects in the environment.
- **The Visual Vocabulary**. From the features, a discrete fixed set of 'words' is obtained, this is known as the dictionary or vocabulary. A single image can then be represented as a histogram over the vocabulary, analogous to bag-of-words for text analysis.
- **Documents and Topics**. The algorithm is given a set of images, and the goal is to discover the objects in the environment. This subsection describes a method known as Latent Dirichlet Allocation (LDA, see Appendix B.1), for object learning.
- **Classification**. Once an LDA model is built, the system can then classify images of new objects. This subsection explains the method for classification, given an LDA model and a new image.

The visual problem of general classification is different when a robot is involved, which has control over the environment. A robotic arm is able to manipulate objects in order to view them from all sides. This section only covers general object classification using LDA, which works well with a wide dataset of objects such as CalTech [Sivic et al., 2005, Fei-Fei and Perona,

2005b]. This section provides a technical review of the pre-requisites for Visual LDA, in particular, describing the method used for classification in this thesis.

6.1.1 Visual Features

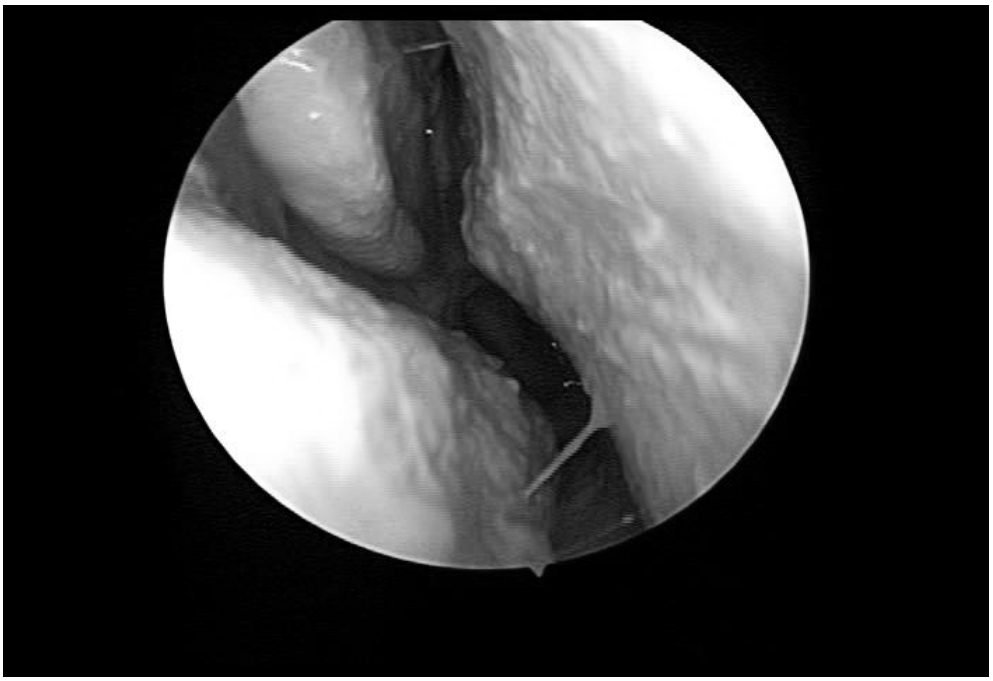
Videos and images as interpreted by cameras and the human eye, contain information about colour and illumination intensity. Although this provides everything necessary to learn about objects in the environment, most of it is redundant. For instance, the amount an object is illuminated does not affect its description, neither does its pose or scale.

One of the most fundamental problems in Computer Vision, is to extract information about an object in an image that is invariant to changes in redundant properties. This area is known as *feature extraction*. The problem was introduced by a well known psychologist David Marr [Marr, 1982], who argued that the underlying biological representation of images is based on the *primal sketch*, the edges of objects in an image. This theory stands to reason, since children usually make line drawings to express the definition of objects, suggesting that this is the way they are represented in the brain.

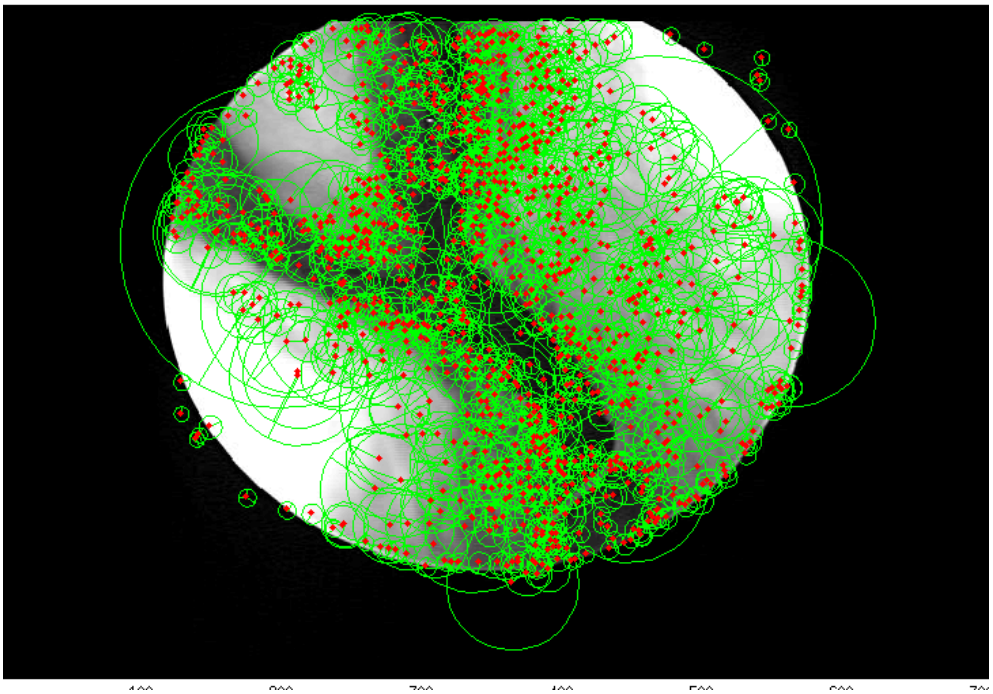
Feature Extraction

The first step towards finding descriptive features in an image is to extract salient points, which are invariant to change of viewpoint and illumination. These feature points or*interest points* are used to identify areas which contain rich information about the objects in the image, and provide the basis for matching and learning algorithms.

A large number of algorithms exist to detect interest points. The most common approach is to find sharp edges or corners in the image. This is done by estimating the spatial derivative or second order derivatives of the image, and finding the local maxima. One of the most well known tools in feature extraction is known as the Laplacian of Gaussians (LoG) operator [Huertas and Medioni, 1986], and works by double differentiating a Gaussian kernel and convolving it with the image, giving an estimation of the second derivative of the image. The parameter of the method is the covariance of the Gaussian, setting it to a higher determinant results in extra blurring of the image, which



(a) Blood vessel



(b) Sift features

Figure 6-1: The process of SIFT extraction, the image is re-sampled at various scales. Strong visual corners are extracted at each scale as shown in Figure 6-1b, the red markers represent the interest points and the radius of the surrounding green circle represents their scale. The figures were generated using SIFT code from [Vedaldi and Fulkerson, 2008].

in turn, finds the corners at a higher *scale*. For example, the wheel on a bike is a corner at a high scale, but the wheel nuts are corners at a smaller scale.

The LoG interest point detector forms the basis of the SIFT feature algorithm. SIFT feature extraction follows the work of [Lindeberg, 1994] who analyse scale invariance in images. Interest points are found in the image by analysing the gradient of the intensity in local regions of the image, sharp intensity changes are taken to be salient points. The use of a Gaussian kernel to smooth the image provides the robustness to change in scale.

There are alternatives to edge and corner detectors when finding interest points in images, one of the most notable algorithms is Maximally Stable Extremal Regions (MSER) [Matas et al., 2004]. This algorithm works by applying a variable threshold to the image in grey scale. This creates a binary image, revealing a number of connected regions in the scene. If the resulting regions are stable (i.e. do not change much) as the threshold varies then they are considered to be interest points. This method is shown to be robust to variation in pose and illumination of objects in the environment, making it a good candidate for a matching or learning algorithm.

The interest points alone are not enough for obtaining information about the scene. The following section introduces *image descriptions* as a means to describe the region local to the feature point.

Feature Descriptors

Identifying points which stay stable across changes in viewpoint and illumination provide the first step towards building robust descriptions of objects. Once the points have been found, it is necessary to describe the region local to the interest point, retaining information about the object it belongs to.

The SIFT algorithm finds 16×16 image patches around the interest points and normalises them to maximise invariance to affine and illumination changes. Although SIFT features have been shown to work well for matching and classification, they have limitations. They are not proven to work in the presence of projective variance, if an object rotates significantly in the environment then the features do not match well. Despite the limitations, SIFT features are robust enough to cope in a wide range of environments, making them a suitable candidate for recognition on a robotic platform.

The resulting SIFT features have 128 dimensions, relatively high when compared to other descriptors. The high dimensionality can cause computational problems, making it difficult to operate quickly on the data. The dimensionality can be reduced using Principal Component Analysis [Jolliffe and MyiLibrary, 2002], or kernel PCA [Schölkopf et al., 1997].

Although SIFT features have become prominent in computer vision for classification and matching, several other feature description algorithms are available. MSER interest points can be described using shape descriptors [Forssen and Lowe, 2007]. A combination of SIFT and MSER features have been used in classification to a proven level of success [Sivic et al., 2005]. Throughout the rest of the thesis standard 128-dimensional SIFT features are used without any dimension reduction, they were chosen because they have recently adapted for use on a GPU [Sinha et al., 2006], allowing a robot to operate in bounded time.

Another alternative is to use dense SIFT patches, where the feature extraction algorithm is neglected and descriptors are taken on a sparse grid across the whole image. This has been shown to work well for understanding natural scene categories, where large parts of the image have coherent descriptions (large patches of grass for example) [Bosch et al., 2007].

6.1.2 The Visual Vocabulary

In order to build a classification system analogous to the text BoW models, a vocabulary needs to be built using the visual features extracted from the images. Words in a document are discrete, an observed word has no ambiguity in its definition. Visual features are drawn from a continuous distribution, however, a small change in illumination, scale or position affects the value taken by the feature. In order to build a visual vocabulary, the space of features needs to be discretised.

The standard way to discretise the space of features, is to take all of the features used in the learning process and cluster them using a suitable algorithm, such as K-means. The centres of each cluster then form the visual 'words'. The visual histograms are then built by iterating through each feature and finding the closest word in the euclidean sense, incrementing the respective

bin by one.

When using an incremental or Online BoW algorithm, not all of the features are available to the algorithm to begin with. So building a vocabulary using K-means is not possible. Using incremental K-means seems like a good starting point, but if the cluster centres change, it will affect the histograms. To put it another way, histograms of the same object under a different visual vocabulary will look different. In the context of text documents, it isn't so much of a problem, the words in the dictionary are fixed. Although the histograms may need to have a larger dimension, the centres do not change. As a result, varying vocabularies are not considered in the Online version of Latent Dirichlet Analysis.

This problem has been studied to a small extent in the Computer Vision literature [Lian et al., 2010, Lazebnik and Raginsky, 2009]. The approach taken in this thesis is to use a fixed dictionary, computed before updating the LDA model. The results in this chapter experiment with the use of a wide dictionary, computed on a large set of images from a publicly available dataset.

6.1.3 Documents and Topics

When using Bag-of-Words for text analysis, a 'document' is well defined, when processing images the definition is not so clear. The most intuitive definition is to use the whole image as a document. Referring back to the text analogy, LDA determines the 'topics'. This requires the topics (histograms of words that always appear together) to move in and out of the documents, in order for the algorithm to differentiate between them. When applied to images, objects are sets of features that coherently move together. The analogy is then that objects are topics.

This works well when the image contains only the object of interest, with some background noise, or when objects are independently moving in and out of the images. This may not be the case for a set of images in front of the robot, where the robot can see all of the objects at all times. The fact that objects have not entered or left the scene makes it impossible for LDA to discriminate between them. A solution to this problem is to segment the image using a low/mid level algorithm and use the segments as documents instead [Cao and

Fei-Fei, 2007]. This generally requires that at least the object (rather than background noise) is contained in the segment, already a well known and difficult problem in computer vision.

The solution to segmentation used throughout Chapters 6 and 7 is a simple object detection algorithm. The algorithm is detailed in Section 6.2, and works by extracting regions (blobs) of the image with a high density of SIFT features. The assumption is that the background has a lower density of features than the object, which is not true in a general environment, but is sufficient for testing the algorithm before employing the more robust segmentation algorithm of Part II. A consequence of using feature density, is that objects which appear close together are placed in the same segment, implying multiple objects per blob. The results show LDA to be robust to this kind of noise, since multiple topics are allowed within documents.

Figure 6-2 illustrates the elements of the model learning process. The rectangular grids represent SIFT features, the cones represent putative objects as detected by the density-based algorithm. Each w_i represents a set of words related to putative object i. The histograms over these words are then used to learn the topics z. Each topic is analogous to an object model, and once learned, they are used to classify images of the objects.

6.1.4 Classification

Once the robot has been given all the training data, and has computed the parameters of the training data, it can classify objects in the environment. The learned parameters of the LDA model give rise to a set of distributions, covered in more detail in Appendix B.1.

In order to classify a new document (or blob), the histograms of words are updated into the Online LDA variational algorithm. In computing the inference, the parameter θ is calculated, which represents the topic proportions for the document. This parameter gives the probability distribution $p(z|\theta)$ for the document, i.e. the probability of a topic, given the document. Assuming that the document contains a single topic, classification can be performed by finding the maximum likelihood of a topic, *i.e.* $z^* = argmax_z p(z|\theta)$. In other words, the object in the image or document is identified as belonging to the

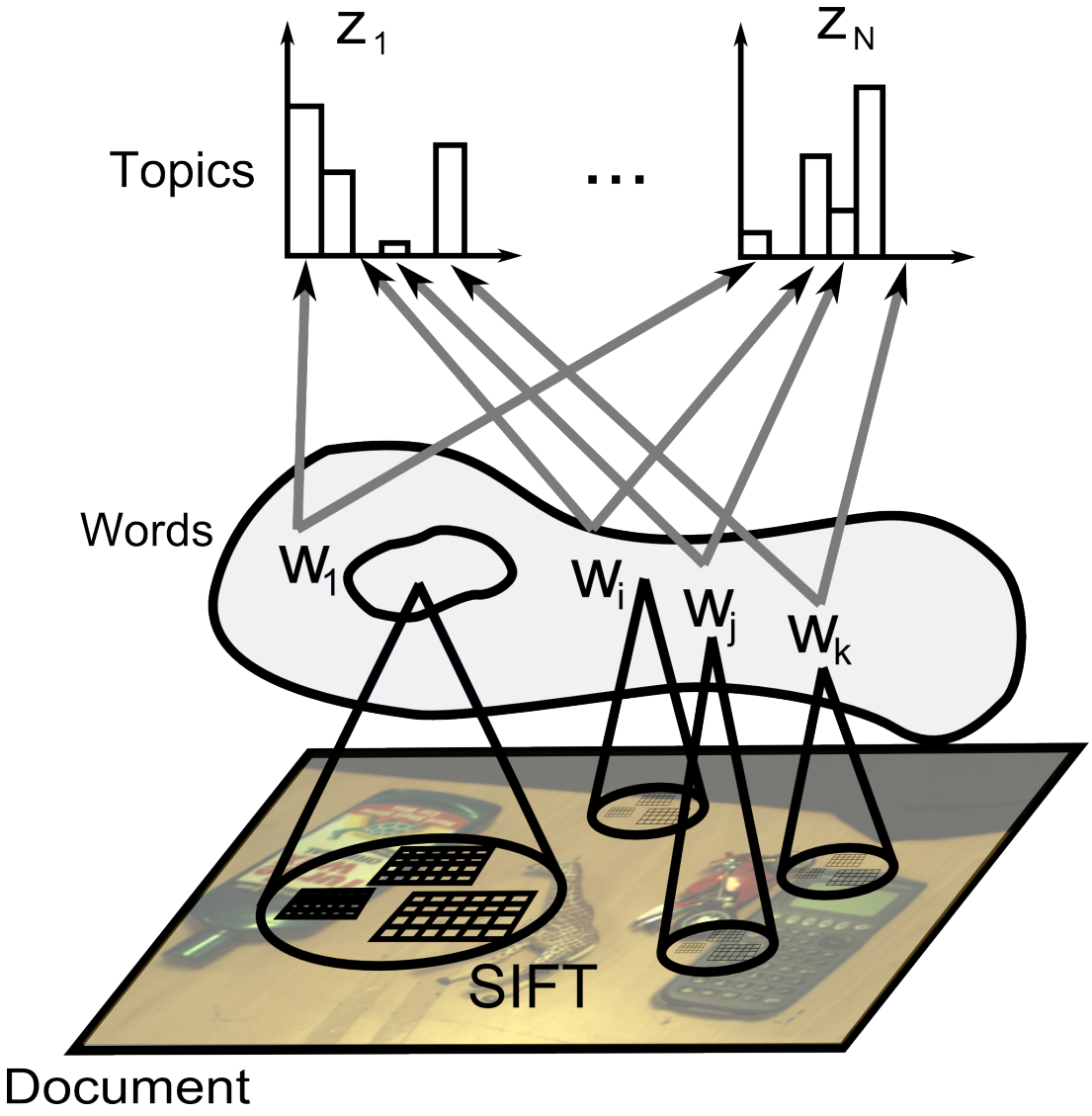
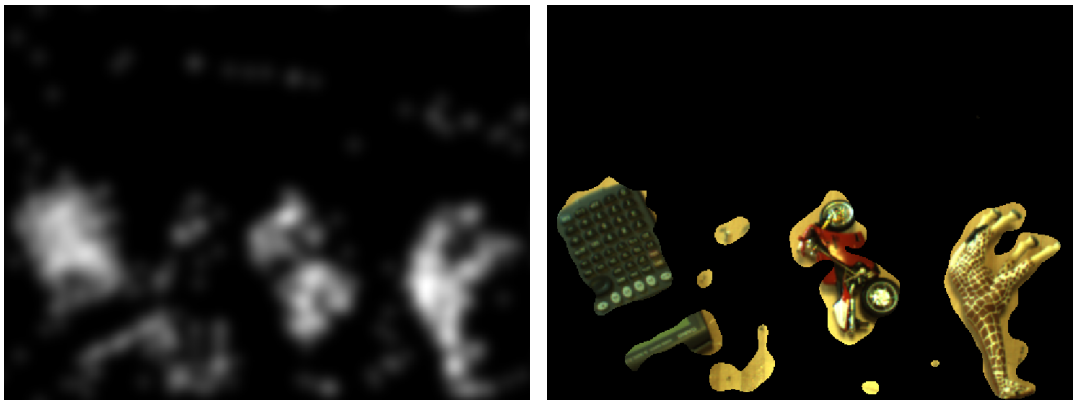


Figure 6-2: This diagram shows the LDA process for visual features. A set of documents and a dictionary of words is given to the algorithm, words are clusters of visual features. Once trained, the topics are found and returned.



(a) Density map (b) Thresholded density map

Figure 6-3: An example of density-based segmentation. The probability map on the left is thresholded to create regions containing putative objects.

class or topic z^*.

For this approach to operate successfully two things must happen, (i) objects must be segmented robustly into documents and most importantly (ii) objects must be represented by single topics.

6.2 Detecting High Density Blobs

The LDA approach to learning, covered in the previous section, required frames of the video to be separated into regions which are likely to contain objects. This section details a method for extracting objects termed 'blobs', based on the density of SIFT features. Regions of the image are more likely to contain an object if they have a higher density of features. This approach makes the assumption that the background has a low feature density, but suffices for an initial set of experiments on the method. The blobs output from this method give the documents needed for the LDA algorithm.

Density estimation can be slow, the usual approach is to use kernel density estimation (KDE) [Scott, 1992]. Given a set of samples $\mathbf{x}_1, ..., \mathbf{x}_N \in \mathbb{R}^K$ the probability density function f is estimated as follows,

$$f(\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} K_f(\mathbf{x} - \mathbf{x}_n) \tag{6.1}$$

Where K_f is a kernel function, usually taken to be a Gaussian or flat kernel. To speed this process up we create a function (image) h that is zero everywhere apart from at the feature points,

$$h(\mathbf{x}) = \begin{cases} 1 : \mathbf{x} \in \{\mathbf{x}_1, ..., \mathbf{x}_N\} \\ 0 : \text{otherwise} \end{cases} \tag{6.2}$$

A Gaussian kernel is then used, denoted N, and convolved with the function h. This process is faster than standard kernel density estimation because convolution is equivalent to multiplication in the frequency domain. Letting H and $\hat{N}$ denote the Fourier transforms of h and N respectively. The convolved function is computed as,

$$(h * N)(\mathbf{x}) = ifft(H(\mathbf{x})\hat{N}(\mathbf{x})) \tag{6.3}$$

Where $ifft$ denotes the inverse fast Fourier transform algorithm. The function $h * N$ is then thresholded at a particular value to extract the regions of highest density. This is a low cost method with a simple implementation, giving rise to a set of putative objects in the workspace.

The use of Online LDA for vision and in a robotics environment provides the first step towards a robot which can learn about the objects around it. The experiments in the following chapter tests the algorithm for practical use on a robot, and also on a publicly available image dataset.

6.3 Experiments in Object Modelling

The theory laid out in the previous sections provides the foundation for a robot to learn object models using Online LDA. This section provides a set of experiments to test the practicality of the methods in a real world environment.

The experiments are motivated by the following hypothesis,

- **Online Learning**. It is possible to effectively use Online LDA for visual learning on a robotic system.

Although Online LDA has been proven to work for text analysis, the extension into vision is not straight forward. Documents are formed using a

dictionary which does not change. A visual word represents a high-contrast patch in the image which is subject to change depending on viewpoint, illumination, and other non-descriptive features. Since Online LDA was built for text analysis, it assumes the use of a fixed dictionary. The first part of the results explores the use of a *Universal Dictionary*, or in other words, a single visual dictionary trained on a wide set of classes. This is an alternative to using an Online dictionary, which changes as new information becomes available. The difference between Online LDA and Batch LDA is also tested showing the Online version to work just as well as Batch.

To summarise the experiments made throughout the chapter,

- **Specific versus Wide Dictionary.** A dictionary is built over specific and wide dictionaries, using the CalTech dataset. This experiment shows that when LDA models are trained using images that weren't used to build the dictionary, the quality in classification is worse than otherwise.
- **LDA versus Online LDA.** It is shown that LDA and Online LDA are of a similar classification quality, advocating the use of online LDA for robotics.
- **Classification using a Robot Vision System**. The method is shown to work for a collection of objects, randomly moved in the robot's workspace.

In total, the results show Online LDA performs well on a robotic platform. Using a wide dictionary shows improved results for a wider set of test objects, advocating its use for Online object learning in robotics.

6.3.1 Specific versus Wide Dictionaries

In this experiment, two dictionaries are developed, a *wide* (W) and *specific* (S) dictionary. Both dictionaries contain 400 words, and both are built from classes in the Caltech 256 database. The wide dictionary is built using 100 different classes, randomly selected. The specific dictionary is built from the following 5 classes: *bikes*, *faces*, *horse*, *guitar* and *starfish*.

The following procedure is adopted to build each dictionary. For each class, a fixed number of images (30) are selected. For each image a fixed number (100)

of SIFT features are randomly selected among a dense grid of locations. This produces a training set of about 300,000 samples from which 400 clusters were learned using k-means (stopping after the evolution fell below 0.1%).

Now we have two dictionaries, we want to compare them in terms of how well an LDA model built over them generalises to include new objects. Two sets, A and B, are formed from CalTech 256. These will be used to build topic models using LDA, and should not be confused with the data used to build dictionaries. Sets A and B are defined as follows; set A ={*a=bikes, b=faces, c=horse, d=guitar, e=starfish* }, and set B = {*f=binoculars, g=gorilla, h=hammock, i=mushrooms, j=air-planes* }. Note that set A is exactly the set used to build a specific dictionary, and set B is a subset of the 100 classes used to build the wide dictionary. Both of the sets are further partitioned in training and test sets. The test datasets comprise of 20 images of each class, the training data are the images left in each class. Figure 6-4 shows a sample of images taken from each class, for each of the training sets.

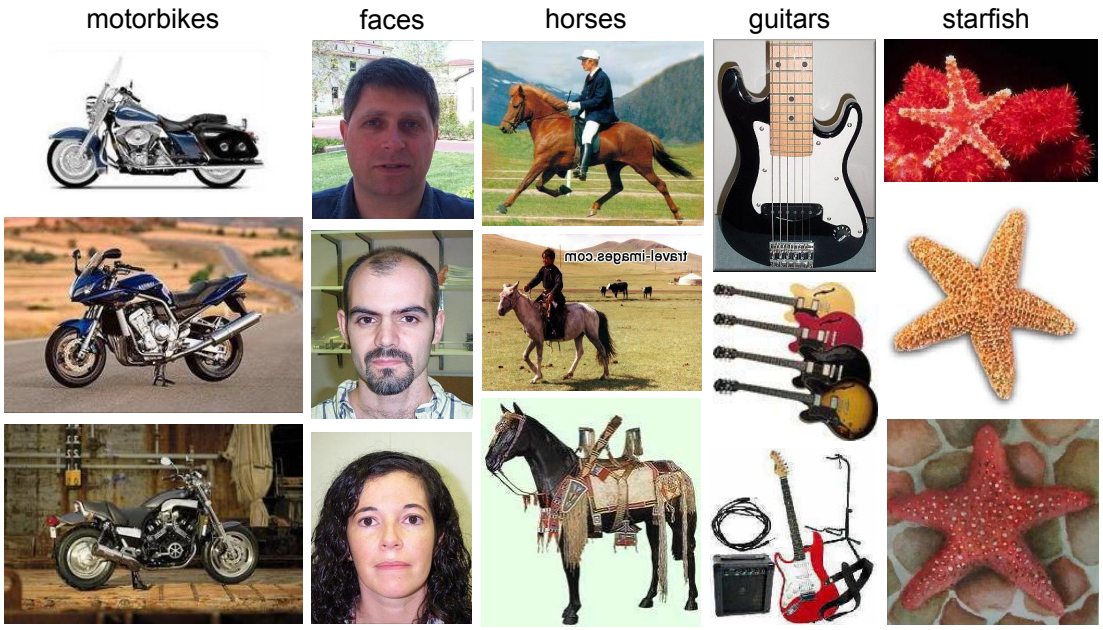
Thus, there are fours sets in total; training set A, test set A, training set B and test set B. Four LDA models are learned; (i) C_{AS} built over the training set A using the specific dictionary (which was drawn from set A); (ii) C_{AW} also built over training set A but using the wide dictionary; (iii) C_{BS} built over training set B using the specific dictionary (taken from set A); (iv) C_{BW} built over training set B using the wide dictionary.

C_{AS}	**a**	**b**	**c**	**d**	**e**
a	9	0	1	10	0
b	0	20	0	0	0
c	0	10	8	1	0
d	0	6	0	14	1
e	0	7	4	6	3

C_{BS}	**f**	**g**	**h**	**i**	**j**
f	15	2	3	0	0
g	6	4	10	0	0
h	5	4	11	0	0
i	7	3	10	0	0
j	10	3	7	0	0

Table 6.1: Using the specific dictionary taken from set A. Confusion matrix C_{AS} has an LDA model trained and tested on set A; 55% accuracy. Confusion matrix C_{BS} has an LDA model trained and tested on set B; 30% accuracy.

The test sets were then used to obtain the classification results in Table 6.1 and Table 6.2. The results in Table 6.1 show that the classification rate is much higher when an LDA model is built with the same images used to build the dictionary. Specifically 55% compared to 30%; it is true neither of these are



(a) Sample images for test set A, used to build the specific dictionary.

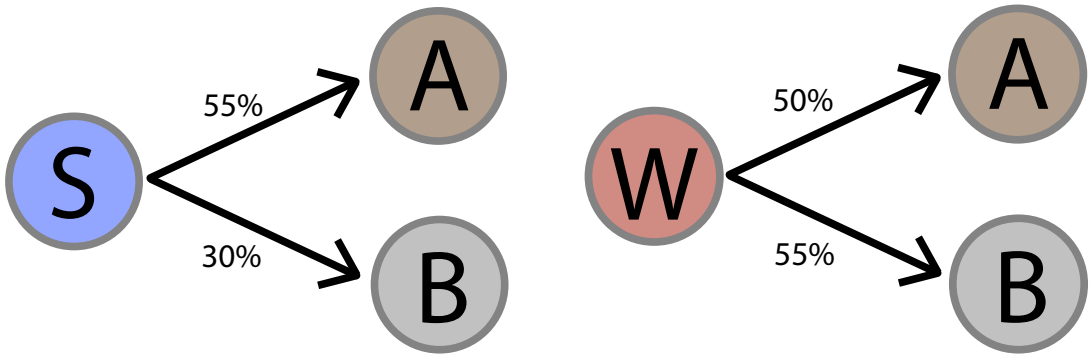
(b) Sample images from test set B.

Figure 6-4: The sample images shown above are taken from the CalTech 256 dataset, test sets A and B respectively. Test set A is used to build the specific dictionary. The images contain objects in a variety of different poses on arbitrary backgrounds.

C_{AW}	**a**	**b**	**c**	**d**	**e**
a	7	0	0	13	0
b	0	19	1	0	0
c	1	2	8	3	6
d	3	3	2	9	3
e	1	3	3	6	7

C_{BW}	**f**	**g**	**h**	**i**	**j**
f	12	1	7	0	0
g	2	9	2	2	5
h	2	0	18	0	0
i	1	2	2	8	7
j	0	4	0	8	8

Table 6.2: Wide vocabulary: Confusion matrix C_{AW}: trained and tested on set A; 50% accuracy. Confusion matrix C_{BW}, trained and tested on set B; 55% accuracy.



(a) Specific Dictionary

(b) Wide Dictionary

Figure 6-5: The graphs shown above describe the experiment to test the use of wide and specific dictionaries. Test set A contains images used to construct S, test set B contains a random set of classes. Dictionary W is constructed on a wide set of classes. Both dictionaries are tested against both sets of images.

impressive but the figure of 55% compares well with the early literature using LDA for visual object classification which is sufficient for this experiment.

The results in Table 6.2 show the classification rates obtained when the wide dictionary is used to train the LDA model. The characteristics to notice here are (i) the classification rates are now about equal for both set A and B, and (ii) these rates are only a little less than the rate for the LDA model for set A when using the specific dictionary taken from set A. Figure 6-5 indicates the results of this experiment, showing the percentage accuracy of confusion matrices using different test sets and dictionaries. It can be seen that the LDA model using a specific dictionary fails when the images are from a larger set.

In summary, this experiment illustrates that using a dictionary learned over a wide variety of object classes is preferable than one that is specific to a subset

of images. This is an important conclusion if robots need to be equipped with a fixed dictionary throughout their training.

6.3.2 LDA versus Online LDA

In this experiment the performance of Online LDA is compared to Batch LDA. The confusion matrices produced using the wide dictionary using Online LDA appear in Table 6.3. These show similar accuracy rates (or better) to the previous matrices (Table 6.2); This experiment concludes that Online LDA is at least as efficient as Batch LDA.

C_{AO}	**a**	**b**	**c**	**d**	**e**
a	14	1	0	5	0
b	0	20	0	0	0
c	2	5	3	3	7
d	6	1	2	6	5
e	6	1	1	0	12

C_{BO}	**f**	**g**	**h**	**i**	**j**
f	10	0	8	2	0
g	0	9	1	2	8
h	1	2	16	1	0
i	0	2	0	7	11
j	0	0	0	1	19

Table 6.3: Confusion matrices using Online LDA. C_{AO}: wide vocabulary, trained and tested on A; 55% accuracy. Confusion matrix C_{BO}: wide vocabulary, trained and tested on B; 61% accuracy.

It is noted that Online LDA out-performs Batch LDA. This is interesting, but no significance is placed on this result as the tests are limited to showing that Online LDA is at least as good as Batch LDA: the data supports that conclusion, and therefore application of interest is tested in the following section.

6.3.3 Classification using a Robot Vision System

This section illustrates how the unsupervised learning method is applied to a robotics scenario. The robot is presented with various objects in its work space. The task is to then move them around in order to learn an LDA and MoT model.

Two videos are used to test the Online LDA method. The first contains three objects, a calculator, bike and bottle, which are randomly moved around the robots workspace. In this video, none of the objects leave the camera's field

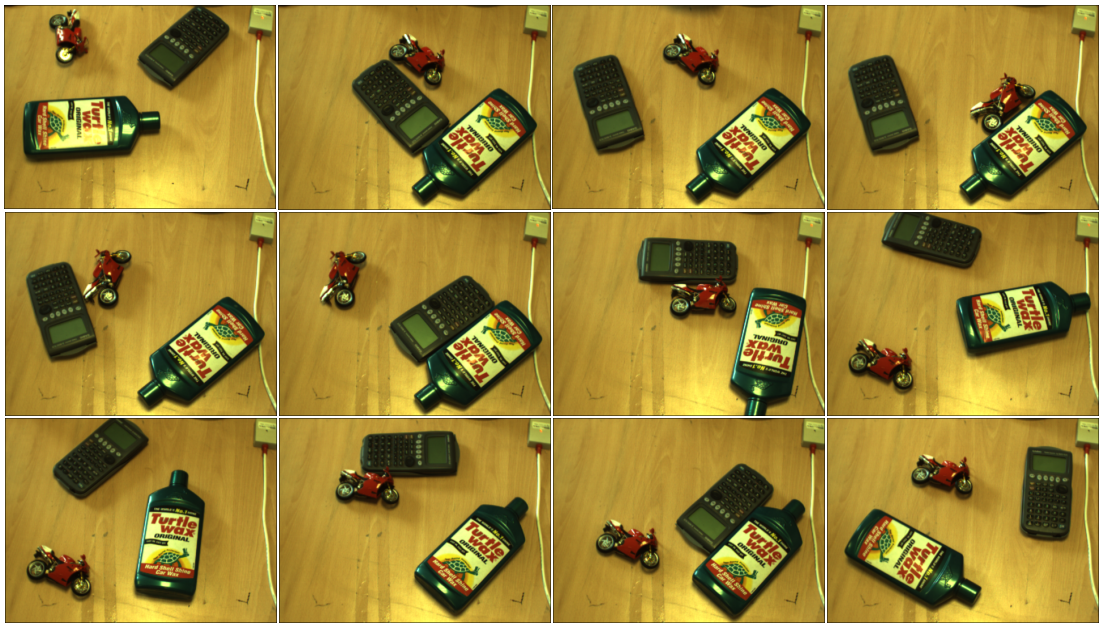


Figure 6-6: The images above are a sample from the sequence used to train the LDA model. The objects are in random poses, and with different reflective properties. Most importantly, objects with high density features appear together, placing them in same blob, and hence, document. LDA is able to classify the objects even when there are multiple objects in a document.

of view. In the second video an extra object is added, a toy giraffe. The objects are placed in the workspace randomly, but this time allowing the objects to leave the camera's field of view and re-entering at random periods of time. For each of the objects we have a ground truth dataset composed of 100 images of the object from different angles on a plain background. Figure 6-6 shows a sample of images from the training set, objects are randomly distributed around the workspace with some moving close together, leading to multiple objects per blob.

To begin with an Online LDA model is built for both of the videos, using mini-batches of 50 frames each. The number of topics is set to be equal the number of objects in the scene. After the training, the learned topics are tested for classification. Results are presented on a confusion matrix which shows the proportion of right and wrong classifications using ground truth data. Each topic z corresponds directly to an object class. In this particular example topic, z_1 is the calculator, z_2 the bike and z_3 the bottle. Table 6.4 shows the confusion

matrix for the first video, the average accuracy is of 83%. The classification shows a relatively large confusion between the bike (z_2) and the bottle (z_3). Using an extra object, and allowing objects to leave the scene, gives a slightly lower overall accuracy of 72%, this is shown in Table 6.5.

O	z_1	z_2	z_3
Calc	94	6	0
Bike	0	100	0
Bottle	0	43	57

Table 6.4: Confusion matrix for three objects, the objects stayed in the robot's field of view for the whole duration of the video. The average accuracy of the matrix is 83%.

O	z_1	z_2	z_3	z_4
Calc	71	0	15	14
Bike	0	19	81	0
Bottle	0	0	98	2
Giraffe	0	0	0	100

Table 6.5: Confusion matrix for four objects, the objects enter and leave the robot's field of view throughout the video. The average accuracy of this matrix is 72%.

Specifying the number of topics in this way (one topic per object in the environment) neglects the fact that, for instance background features, may become a topic of their own. Also, in most real situations, a robot will not know the number of topics of objects it will encounter a priori.

To assess how the algorithm converges, the *perplexity* is plotted against each iteration, as in Hoffman's original paper [Hoffman et al., 2010a]. Figure 6-7 shows the results of this experiment. The algorithm seems to converge quickly and settle at a value of 20. This measure of convergence would allow an autonomous robot to decide when it has learned enough about the objects in the environment, a halting criterion for the robot.

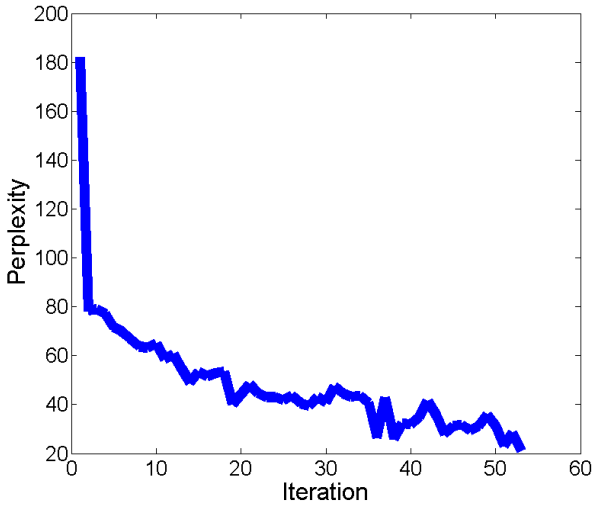


Figure 6-7: Perplexity against iterations of Online LDA.

6.4 Discussion

Online LDA can be used for vision in robotics, allowing the robot to update object models. The results show that a good, wide, dictionary suffices for Online LDA in computer vision; without the need to update the parameters of the model, or the dictionary for new data.

A limitation of the model is that it requires a fixed number of topics to be specified beforehand. The results show that the number of topics significantly affects the quality of the classification results. Thus, choosing the correct number is essential for learning about an arbitrary number of objects. This observation also leads to the problem of merging multiple LDA models. If the robot learns a set of topics from several different scenes independently, the models need to be merged, identifying similar objects between models and updating the total number of topics.

The success of the algorithm is heavily dependent on segmenting the scene into potential objects. We use the density of SIFT features to decide what constitute putative objects. This is a simple and effective method, but if the background contains densely clustered features then the algorithm will fail. This can be avoided by using a different segmentation algorithm, but for simplicity of implementation, this was not considered here. The use of a more advanced segmentation algorithm is investigated in Chapter 8.

Despite these limitations, we have shown that Online LDA is a useful tool for autonomously learning about objects on a robotic platform. The following chapter goes on to extend LDA for improved classification results, and describes an algorithm for manipulating objects in the environment.

Chapter 7

Mixtures of Topics and Self-Supervision

The previous chapter of the thesis introduced Latent Dirichlet Allocation as a method for learning on a robot. It was shown the method works Online with the use of a suitable dictionary. Although Online LDA is a necessary requirement for the robot to start learning, it is not directly in support of the hypothesis. The work carried out in this chapter explores the use of robotic manipulation in the learning process.

Two main contributions are made in the chapter:

- A new **Mixture of Topics** (MoT) model is introduced to represent objects in the robot workspace. The model is shown to outperform LDA when used on robotic data, and shown to be robust to data from wide object classes using the CalTech 256 dataset. The MoT model requires that objects are first discriminated from their background, and that the data is labelled. This motivates a need for object segmentation, covered by the second contribution of the chapter.

- The second contribution of the chapter, is a method for **self-supervision** with a robotic manipulator. The robot then removes other objects from the environment, and focuses on an object of interest, viewing it from multiple angles. Once a model is built, other objects in the environment can easily be distinguished and separately learned about.

The work is in support of the hypothesis, since planned robotic motion is

used to move objects in the environment. The standard method for learning object models is to provide the algorithm with a set of training images from a wide dataset, such as the web. Using a robot to manipulate the objects allows them to be viewed from multiple sides. This improves the visual model by providing information of objects from more than one view, widening the variance, and improving recognition and classification.

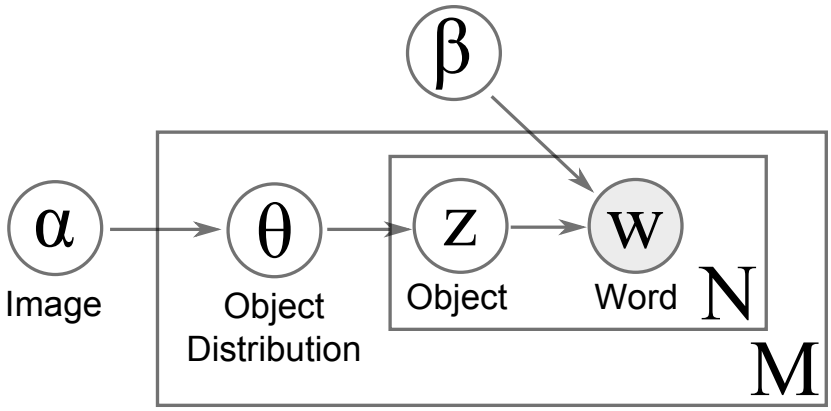
The chapter begins by introducing the Mixture of Topics model as an extension of LDA, and then goes on to describe the process of self-supervision, and how the robot plans its movement in the environment. Motivation for using Mixture of Topics over standard LDA is shown, along with improvements in classification quality on a robotic manipulator. Finally, the model is tested using CalTech 256 to show robustness to classes containing a large number of different objects.

7.1 A Mixture of Topics

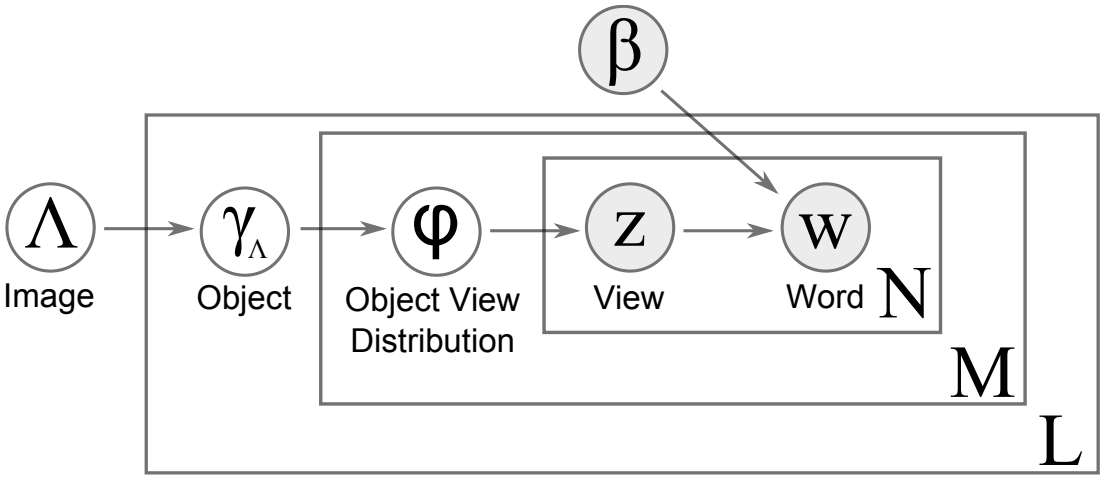
The classification method shown in Chapter 6, assumes that documents are the images themselves, and topics are objects within the images. This works relatively well on a wide class of objects, but if a particular object significantly changes visual appearance depending on its view, then it could be considered as a different object entirely. An example of this are the classes 'car' and 'motorbike'. From the side both look similar, appearing to contain two wheels and a body. When viewed from the front the objects are much different however, giving strong evidence that their visual category is dependent on view.

An alternative view is to take the *objects* as documents, then each view is a distribution of topics. This can be modelled by adding an extra variable into the graphical LDA model. At the top of the hierarchy are the images, which contain objects, which contain views of the objects, which contain words. This model is shown in Figure 7-1b, and is named 'Mixture of Topics'. Writing the graphical model as a probability distribution and using variational Bayes, it would be possible to derive a fully Online method for the Mixture of Topics model. Using the same technique as in [Hoffman et al., 2010a] for LDA.

Once an LDA model is trained on a set of data, new images are passed to the variational algorithm, returning a distribution of topics. The LDA process



(a) The 'Latent Dirichlet Allocation' graphical model.



(b) The 'Mixture of Topics' graphical model.

Figure 7-1: Comparing the Mixture of Topics model to standard Latent Dirichlet Allocation. The greyed out boxes are the observed variables. A Mixture of Topics model has an extra level to the hierarchy representing an image or video as containing a number of objects, with multiple views. Standard LDA represents an image as a set of objects with only one view.

then involves choosing the topic of maximum likelihood in order to classify the data. Looking at the topic distributions for 4 objects placed in front of the robot, show in Figure 7-3, it is clear that some objects have a high probability of containing more than one topic. For example, Figure 7-3c shows the bottle to have a high probability of containing topics 3 and 5. This may be due to different views of objects correlating to topics, or also, groups of features which are consistent between objects. For example, a wheel appears on both a bike and a car making it a topic of its own.

Instead of choosing a maximum likelihood topic, a model can be built using the topic distributions alone. Each piece of data given to LDA translates in to a vector of probabilities, the probability that the data belongs to a topic. Assuming these vectors are drawn from a Dirichlet distribution, a model is built for each object independently. This can be done in a supervised manner, using Bayes Theorem, or unsupervised with the addition of a clustering algorithm. When using a robot, it is possible to *self-supervise* the process, and is discussed in more detail in Section 7.2.

The distribution of topics is assumed to be sampled from a Dirichlet distribution. The problem is to find the probability of the parameters of the model ω, given the topic distribution of a new object θ,

$$p(\omega|\theta) \propto p(\theta|\omega)p(\omega) \tag{7.1}$$

$$= \text{Dir}(\theta|\omega)p(\omega). \tag{7.2}$$

To train the model, the distribution $p(\theta|\omega)$ is found for the set of training data $\theta_1, \theta_2, ..., \theta_N$. The parameters ω of this Dirichlet distribution are calculated using [Minka, 2003]. The number of dimensions of ω corresponds to the number of topics chosen in the LDA algorithm. If labelled data is available then each ω is computed separately for each known class of objects. Alternatively, the Mixture of Topics can be used in an unsupervised setting. The separate classes are found using a clustering algorithm, such as k-means, on all of the θ_i.

The model can be modified to perform online by sampling the old distribution, appending new data to the generated samples and recomputing the model parameters (ω), as described above. This method for Online MoT is tested in Section 7.3.3, (termed *online-unsupervised*).

The graphs in Figure 7-3 show the topic distributions of objects under a 3D projective transformation, for four separate objects. It is clear from the distributions that the different objects are separated across each of the 8 dimensions. Figure 7-2 shows these distributions projected on to their largest principle components. The classes are well separated in the two dimensions, using a discriminative clustering algorithm such as k-means is appropriate for determining the classes, with minimal loss in quality.

The MoT model can be described simply as a Bayesian model over the topic

distributions returned by LDA. Different views of the object are given to the algorithm in order to build a model of the whole object. This requires that the documents given to MoT contain only the object of interest, and that the robot is able to manipulate objects so they are viewed from multiple angles. These problems are addressed in the following chapter.

7.2 Self-Supervised Learning

The Mixture of Topics model defines documents as objects. This means that the algorithm needs to be given images of different views of whole objects in order to work effectively. This is in contrast to LDA, where documents are images containing multiple objects. Object segmentation is one way to ensure that documents contain single objects, but requires assumptions to be made about the objects.

This section explores the use of robot motion to remove other objects from the camera frame, and also manipulate an object of interest so that it is viewed from multiple angles. This simplifies object segmentation when the background is plain, since objects can be assumed to have a higher density of visual features. A combination of motion based object segmentation and self-supervision is explored in Chapter 8.

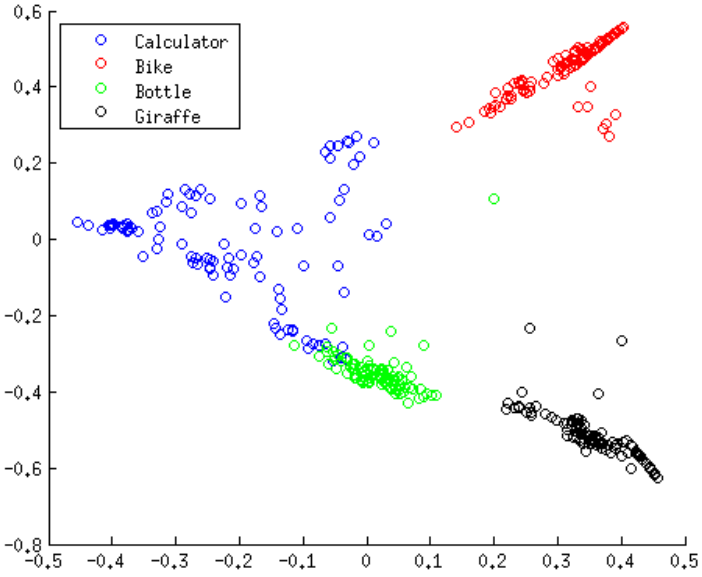
Since the robot has control over the environment, it is able to make changes to objects in the scene in order to benefit its learning. Classical learning algorithms can be separated in to two distinct categories:

- **Supervised**. Where the robot has access to labelled data. Algorithms of this type need a supervisor to identify the objects within an image, in order for the algorithm to learn the classes.

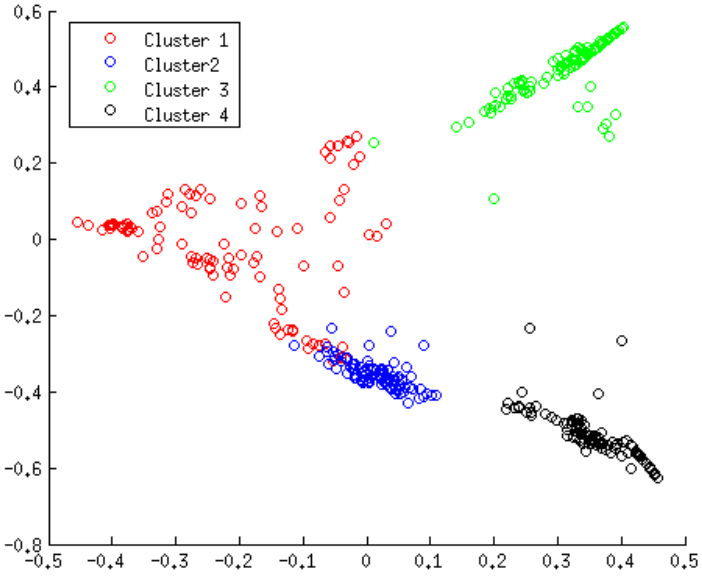
- **Unsupervised**. Where the robot generates its own labels given only the data. Usually some extra prior knowledge is given to these algorithms, such as the number of classes, but this is not necessary in principle.

The use of a robotic manipulator allows enough control over the environment to build a classifier that can label its own data, and is termed:

- **Self-supervised** learning. This approach trains a supervised classifier, in the machine learning sense, using a robotic manipulator to obtain

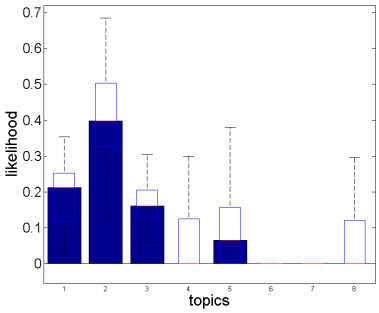


(a) Supervised

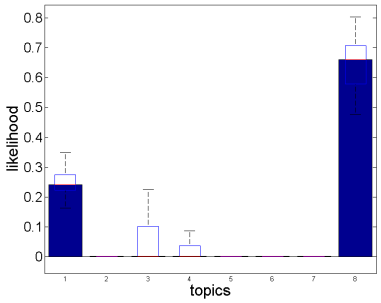


(b) Using kmeans (95% correct)

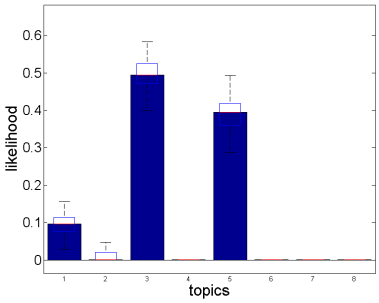
Figure 7-2: The topic distributions projected on to the 2 largest principle components. Graph (a) shows the distribution of each class and graph (b) the result of applying k-means to the data.



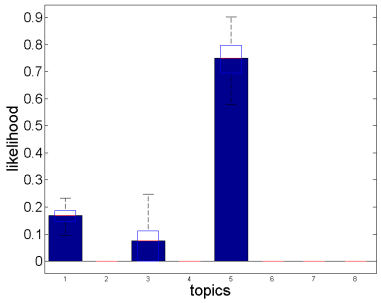
(a) Calculator



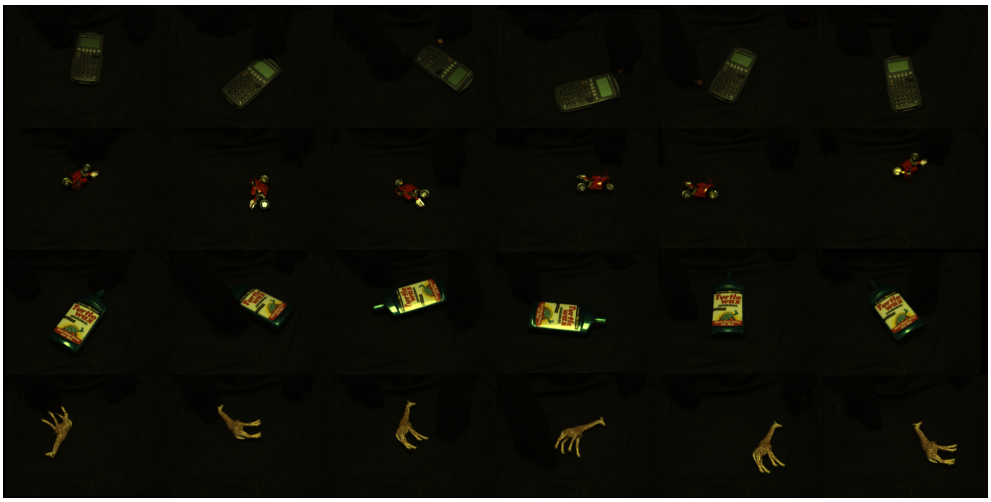
(b) Bike



(c) Bottle



(d) Giraffe



(e) Ground truth data

Figure 7-3: The distribution over the topics of 4 ground truth objects. Given the LDA model trained on objects entering and leaving the scene. Subfigure 7-3e shows a sample of the ground truth images used to generate the topic distributions in Subfigures 7-3a - 7-3d

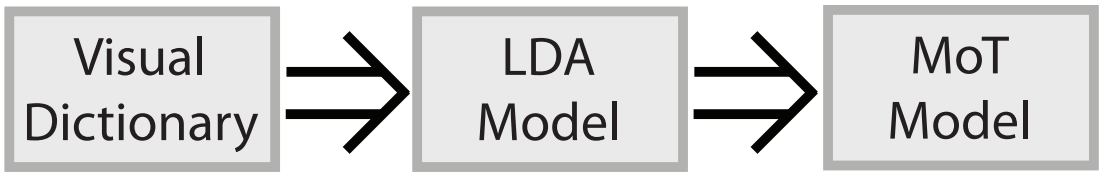


Figure 7-4: The order in which information is built in the Mixture of Topics model. A visual dictionary is generated using a set of data, histograms are built over this in order to form the LDA model. Images of objects are then passed to LDA in order to generate data to build a Mixture of Topics model.

multiple views of the single object. The method makes the assumption that the robot is holding or pushing the same object until it chooses to move on to another. A supervised Bayesian classifier is built over the data. The robot then chooses another object with the lowest probability of fitting any previous models, and begins to manipulate it instead.

The algorithm is only given the number of objects it has to learn about, without any labelled data. In a global sense, the process of self-supervision turns a supervised classifier in to an unsupervised one, inheriting all of the benefits of supervised learning, assuming that there is control over objects in the environment.

The algorithm uses a Mixture of Topics model as the supervised classifier, in this model it is assumed that objects are a mixture of different topics. Figure 7-3 shows the distributions of 4 objects viewed around 360 degrees on a plane. The LDA model is set to 8 topics. Although each object has an individual maximum likelihood topic, there are other are other topics in the distribution with a relatively high probability.

All of the theory needed to manipulate and learn about objects in a workspace has been covered. The full algorithm for automated learning on a robot is broken into stages.

1. The first step is to find the visual dictionary. LDA requires that the dictionary is built independently of the distributions in the statistical model. So to ensure that the algorithm works incrementally, a universal dictionary can be employed instead. This approach builds a single dictionary, over a large publicly available image dataset. The alternative is to get the robot to manipulate the objects in front of it first, building the

dictionary before the LDA model. The pros and cons of using a universal dictionary compared to a specific one are discussed in the Chapter 7.

2. In order to build a Mixture of Topics model, an LDA model is required first. The robot could recursively and randomly push objects in the environment through their centre of mass; using the method laid out in Section 6.2. The model could also be trained separately on a wide dataset of images, opening up the possibility of a universal LDA model. For the experiments in this section the same data is used as in Chapter 7, so that the results of classification using LDA can be compared with self-supervised MoT.

3. Once an LDA model is built, uses self-supervised learning is used to build a Mixture of Topics model. Given the number of objects in the environment K, the robot,

   (a) Chooses an object (blob) at random.

   (b) Pushes all other objects to the edge of the visual field.

   (c) Recursively pushes the object of interest, while building an unsupervised MoT model.

   (d) The objects are presented to the robot again. It chooses the object *least* likely to belong to previously learned models and repeats from (b).

Since the algorithm works in stages: Building a dictionary, finding the LDA model before finally building the Mixture of Topics model; it is not an Online or incremental method. Improving Online LDA so that a changing vocabulary is possible is a current area of research, and the use of a universal dictionary was discussed in Section 6.1.2. The separation of modules is summarised in Figure 7-4, learning the dictionary, LDA model and MoT are separated. Changing a model to the left, implies reprocessing all of the data for modules on the right. Online Mixture of Topics with a changing LDA model is not currently possible but a good direction for future research in the area.

7.2.1 Robotic Planning

To manipulate objects in the environment, a simple planning algorithm is employed. The robot is calibrated to the cameras as detailed in Section 3.1. The objects are constrained to a table and points are defined for the robot to move to, in camera coordinates, and the robot is constrained to move in straight lines. Objects are represented by their feature density as defined in Section 6.2. There are two problems to be solved by the planner, firstly the robot needs to be able to push the object through its centre of mass; secondly it needs to avoid pushing other objects in the environment.

Object Pushing

When the robot chooses an object to manipulate there are three key points given to the planner, the centre of mass $\boldsymbol{p}_c$ computed as the average feature on the object, the target position $\boldsymbol{p}_t$ and the point the robot starts at $\boldsymbol{p}_s$. The hull that defines the boundary of the object is also given to the algorithm.

The robot is constrained to move in straight lines, so the first problem is to parametrise the line for the path of the manipulator, as follows,

$$\boldsymbol{f}(t) = \boldsymbol{p}_s + t\boldsymbol{n} \tag{7.3}$$

Where $\boldsymbol{n} = \frac{\boldsymbol{p}_s - \boldsymbol{p}_t}{||\boldsymbol{p}_s - \boldsymbol{p}_t||}$. When $t \in [0, 1]$ the function $\boldsymbol{f}$ defines a line between the start point and the target position. We want to push the object so that the centre of mass lies over the target.

Assuming that we know the distance from $\boldsymbol{p}_s$ to the edge of the object, call this distance d. The robot take a path along $\boldsymbol{f}$ from $t = 0$ to $t = d + ||\boldsymbol{p}_c - \boldsymbol{p}_t||$. To calculate d Algorithm 4 is used. Two hull algorithms are assumed, findConvexHull and containedInConvexHull, both of which are standard in computational geometry [Barber et al., 1996].

Once the path has been found in the camera space. The calibration and inverse kinematics converts the positions into motor positions and velocities, more details on this can be found in Section 3.1.

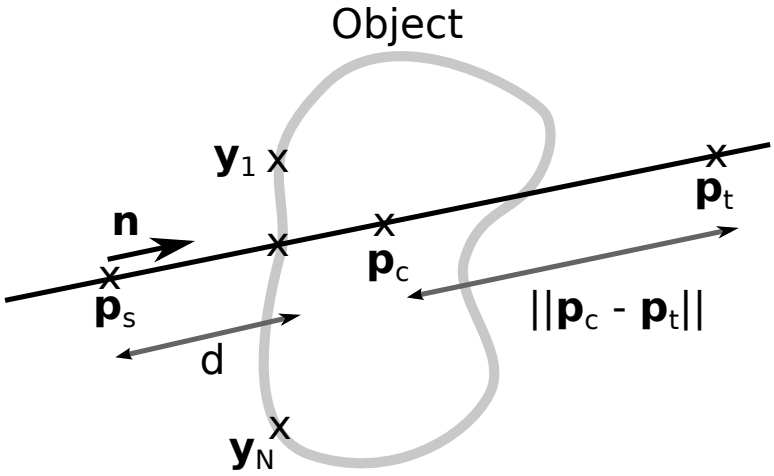


Figure 7-5: Geometry of the motion planner.

Algorithm 4 Algorithm to find the distance to the edge of the object

Require: $\boldsymbol{p}_s$, $\boldsymbol{p}_t$, $\boldsymbol{p}_c$, $\boldsymbol{x}_1$, ..., $\boldsymbol{x}_N$
  $t \leftarrow 0$
  stop $\leftarrow 0$
  $[\boldsymbol{y}_1, ..., \boldsymbol{y}_K] = \text{findConvexHull}(\boldsymbol{x}_1, ..., \boldsymbol{x}_N)$
  while stop $= 0$ **do**
    stop $= \text{containedInConvexHull}(\boldsymbol{f}(t), [\boldsymbol{y}_1, ..., \boldsymbol{y}_K])$
    $t \leftarrow t + \varepsilon$
  end while

Object Avoidance

To avoid other objects in the environment, a probabilistic approach is taken. We seek the probability that a particular path will cause the manipulated object to encounter another object in the environment.

To achieve this, the probability of the function $\boldsymbol{f}$ given the hull parameters $\boldsymbol{y}_1, ..., \boldsymbol{y}_K$ is found by marginalising over the variable t,

$$p(\boldsymbol{f}|\boldsymbol{y}_1, ..., \boldsymbol{y}_K) = \int_{t=0}^{d+||\boldsymbol{p}_c - \boldsymbol{p}_t||} p(\boldsymbol{f}|t, \boldsymbol{y}_1, ..., \boldsymbol{y}_K) p(t|\boldsymbol{y}_1, ..., \boldsymbol{y}_K) \tag{7.4}$$

$$= \int_{t=0}^{d+||\boldsymbol{p}_c - \boldsymbol{p}_t||} p(\boldsymbol{f}(t)|\boldsymbol{y}_1, ..., \boldsymbol{y}_K) \tag{7.5}$$

Where the probability distribution $p(\boldsymbol{f}|\boldsymbol{y}_1, ..., \boldsymbol{y}_K)$ is the density of sift

features described in Section 6.2, with everything inside the hull set to zero.

The maximum of the probability density function in equation 7.5, is found by choosing a set of potential points. There are certain constraints to be considered, for example, a trivial solution to the optimisation problem is to have a zero length path where the object is not moved at all. The vectors $\boldsymbol{p}_c$, $\boldsymbol{p}_s$ and $\boldsymbol{p}_t$ are all constrained to be a minimum distance apart. The set of start and end points are randomly sampled, and the maximum is chosen. A more advanced optimisation algorithm such simulated annealing could improve or further refine the correct trajectory to take, but the computation is fast enough to operate well, with a reasonably dense initial sampling.

7.3 Experiments

Three experiments are made in this section, to test the applicability of an MoT model in robotics, and to verify the algorithm on a dataset with classes containing a large variety of objects:

- **Motivation for Mixture of Topics.** The experiments made in this section show the topic proportions that LDA returns, and how a single object can contain more than one topic. This motivates the need for a model which describes objects as multiple topics, rather than one object per topic, as in standard LDA.

- **Self-supervised Topic Mixtures.** The MoT model is tested with robot data, using a robot which uses self-supervised learning. The results outperform the standard LDA results given in the previous chapter.

- **Verification using CalTech.** The Mixture of Topics model is tested on the CalTech dataset. This shows that the model works well on classes with a high intra-class variance, i.e. on objects of the same class in a large variety of different poses, and locations.

The chapter concludes with a method for learning about objects in the environment autonomously. Improving results above standard visual LDA, and shown to perform well on classes containing a large number of different objects.

7.3.1 Motivation for using Mixtures of Topics

The results presented in this section provide some motivation for using a Mixture of Topics model rather than LDA. The main argument is that when the number of topic in LDA is greater than the number of objects, certain objects have a high likelihood of containing more than one topic. This idea was introduced in Section 7.1, particularly in Figure 7-3, where the topic distributions are shown for a collection of ground truth objects.

When using LDA for classification, there is motivation to increase the number of topics beyond the number of known objects. This technique was used in [Sivic et al., 2005] to reduce the negative effects of background noise. The results shown throughout the rest of this section prove the same result, while demonstrating the multiple topic per object problem.

The test videos used in this section are the same as for Chapter 6. Two videos of objects moved in the robots workspace, comprising the objects 'Calculator', 'Bike', 'Bottle', and 'Giraffe'. In the next experiments the number of expected topics is chosen as 8. Table 7.1 shows results from the first video. Some of the topics do not match any of the ground truth data, this implies that they are topics which were part of the training set, but not in the ground truth. Background noise and motion blurred features are a likely cause. The bike data is separated in to two topics, z_2 and z_7. If these topics are merged, the overall accuracy of the confusion matrix (neglecting empty topics) is 88%.

O	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8
Calc	0	0	0	0	0	100	0	0
Bike	2	55	0	0	0	33	10	0
Bottle	100	0	0	0	0	0	0	0

Table 7.1: Confusion matrix for three objects, the objects stayed in the robot's field of view for the whole duration of a 1000 frame video

Increasing the number of topics for the second video produces similar results, as illustrated in Table 7.2. Again the objects are a mixture of different topics in the model. For example, the calculator is attributed to topics z_1, z_2 and z_4, merging these topics and neglecting empty ones yields a 4×4 matrix with 92.5% accuracy. In both cases the accuracy is improved by choosing a higher number of topics than there are objects. Allowing extra topics for

O	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8
Calc	10	69	3	1	9	0	0	8
Bike	0	0	0	0	0	0	0	100
Bottle	0	0	90	0	9	0	0	1
Giraffe	0	0	0	0	100	0	0	0

Table 7.2: Confusion matrix for four objects, the objects enter and leave the video throughout

background items lowers the total amount of noise contributing to the minimal number of topics used for Tables 6.4 and 6.5.

As described in Section 6.3.3, images are classified by maximising the probability distribution $p(z_i|\theta)$ where θ is the test document. Figure 7-3 shows a graphic of the topic distributions for each ground truth objects. As it can be seen single objects are a mixture of topics rather than a single one.

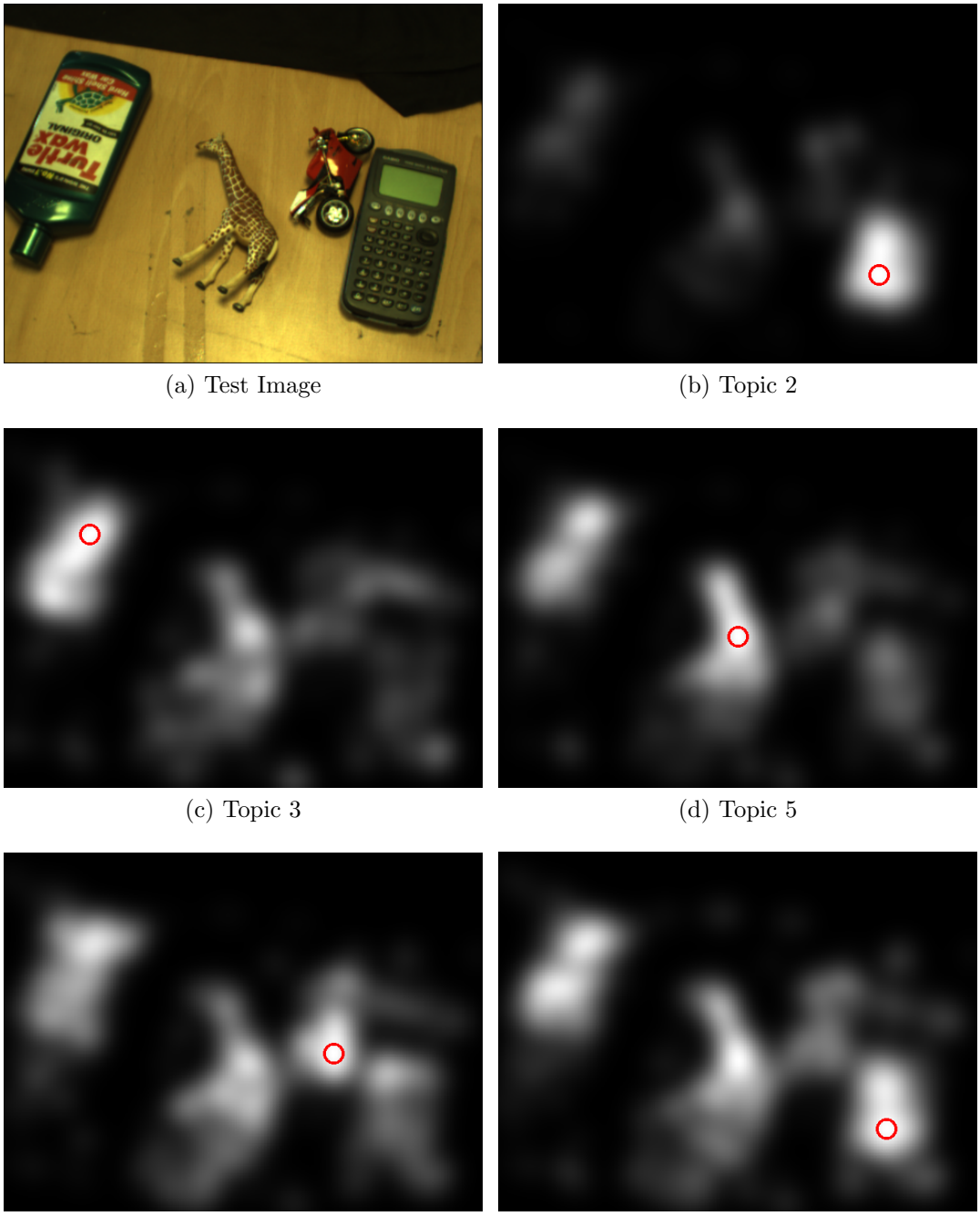
The LDA algorithm not only gives the distribution of topics for a particular document, but also the distribution of words. This distribution can be used to find words in a document which correspond to a particular topic. For example, what words make the bike topic. Given a new document $\hat{\theta}$ and topic z_i, the probability distribution $p(\mathbf{y}|z_i, \hat{\theta})$ is sought for a particular descriptor $\mathbf{y}$. Marginalising over each word w_j gives,

$$p(\mathbf{y}|z_i, \hat{\theta}) = \sum_{j=1}^{N} p(\mathbf{y}|w_j, z_i, \hat{\theta}) p(w_j|z_i, \hat{\theta}) \tag{7.6}$$

The distribution $p(w_j|\hat{\theta})$ is given by LDA and $p(\mathbf{x}|w_j, \hat{\theta})$ chosen as a Gaussian. To build a probability map over the whole image, for a particular pixel $\mathbf{x} \in \mathbb{N}^2$ we let,

$$p(\mathbf{x}|z_i, \hat{\theta}) = \sum_{k=1}^{K} N(\mathbf{x}|\mathbf{x}_k, \Sigma) p(\mathbf{y}_k|z_i, \hat{\theta}) \tag{7.7}$$

Figure 7-6 shows the results of this experiment for the topics which most correspond to the ground truth objects. The circle in the figures shows the most likely point as calculated by Equation 18. It is clear that the topics correlate well with the objects in the test image and the results of the confusion matrix



(a) Test Image (b) Topic 2

(c) Topic 3 (d) Topic 5

(e) Topic 8 (f) Topic 7

Figure 7-6: Distributions over descriptors given the test image (7-6a). The red marker displays the region of the image most likely to belong to the topic.

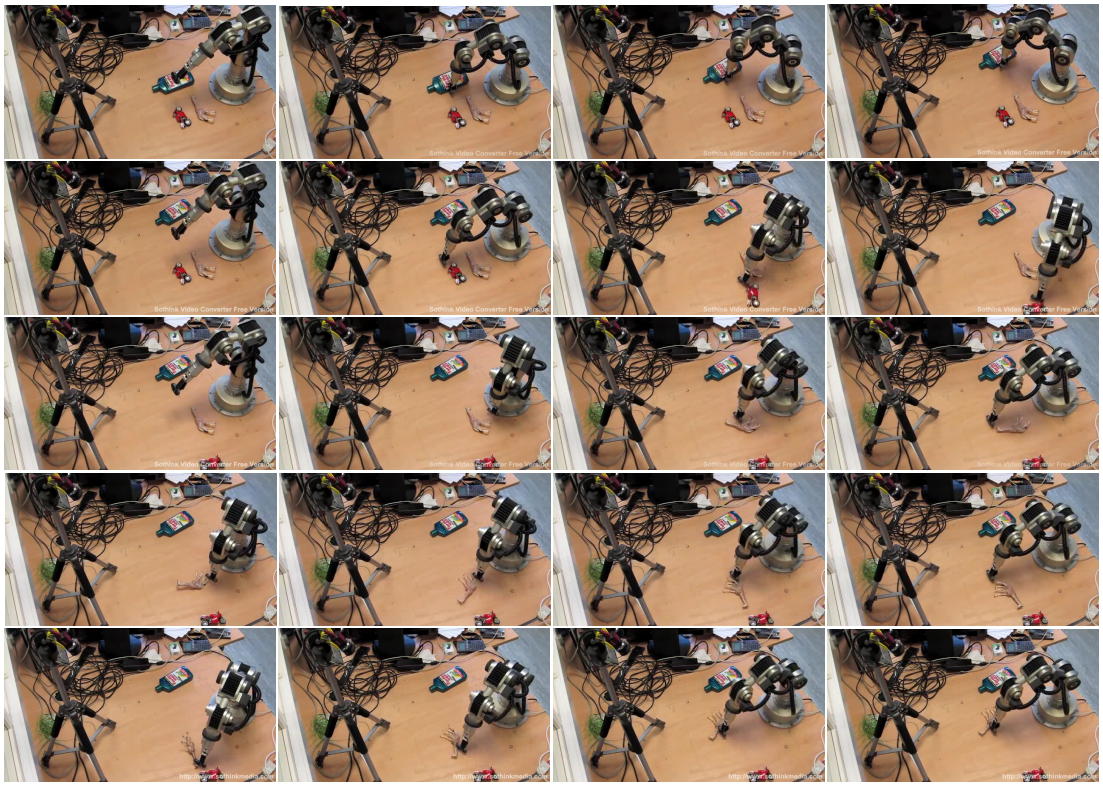


Figure 7-7: Frames of the robot manipulating a toy giraffe. The robot first removes objects in the workspace which are likely to belong to a model it has already learned about, before recursively pushing a giraffe in the scene.

in Table 7.2. For example topic z_2 corresponds to the calculator and this can also be seen to be the case in Figure 7-6c.

The probability map in image 7-6f is from a topic which does not match any of the objects in the confusion matrix. The result is a distribution of descriptors which are close to uniformly distributed over each of the objects. All of the probability maps exhibit noise from each of the objects, but the point of maximum likelihood is a good match.

To summarise, the results show that when the number of topic is chosen to be larger than the number of objects; the quality of classification improves, since background noise becomes a topic of its own. Also, objects can be represented by multiple topics, suggesting that topics relate to specific views rather than the objects themselves.

7.3.2 Self-Supervised Topic Mixtures

The results in this section experiment with Mixtures of Topics, self-supervision, and whether it makes an improvement over LDA alone. In the experiment, the robot is given a set of 4 objects, as in Table 7.2. A single object is isolated from the group and pushed on the table. This process is repeated until all of the objects have been learned about. In other words, until the robot has played with each object individually. Figure 7-7 shows sample frames of the robot performing the self-supervision task. The robot has already learned about the bottle in this video and it selects the object least likely to belong to that class, in this case the giraffe. The robot goes on to manipulate the object, before moving on to the bike.

	Calc	Bike	Bottle	Giraffe
Calc	95	5	0	0
Bike	0	100	0	0
Bottle	0	1	99	0
Giraffe	0	2	7	91

Table 7.3: Confusion matrix for four objects, built using a Mixture of Topics model. The accuracy of this matrix is 96%.

O	Calc	Bike	Bottle	Giraffe
Calc	71	0	15	14
Bike	0	19	81	0
Bottle	0	0	98	2
Giraffe	0	0	0	100

Table 7.4: Confusion matrix for four objects using standard LDA. The accuracy of this matrix is 72%.

The confusion matrix shown in Table 7.3 shows the results of classification against a ground truth dataset of each object on a plain background. There is a significant improvement over the standard LDA model shown in Table 7.4, with an accuracy of 96% rather than 72%. These results demonstrate the value of self-supervision when learning a Mixture of Topics model for objects.

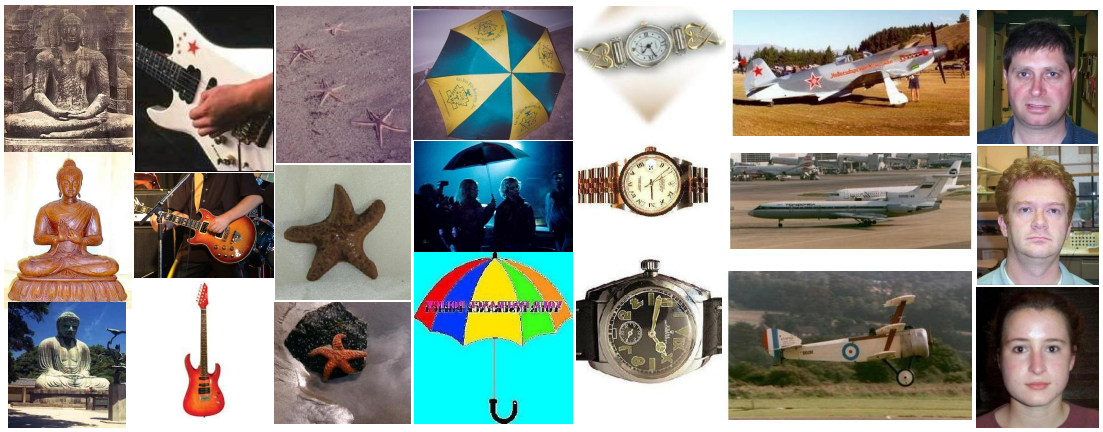


Figure 7-8: A selection of images used for training the Mixture of Topics model. Each column represents a different class from the CalTech 256 dataset.

7.3.3 Verification using CalTech

The results in the previous sections of this chapter show that a Mixture of Topics model works well for a small number of specific objects. This section tests the Mixture of Topics model on the CalTech dataset [Griffin et al., 2007]. The aim is to verify the use of a MoT model on a larger set of object classes, with a high inter-class variance.

The images from the dataset are split in to *training* and *test* sets. The former set is used to build the object model with, and the latter to test the models. Supervised and unsupervised Mixture of Topics is compared on the CalTech dataset. Finally, the Dirichlet estimation described in Section 7.1 is extended so that it can estimate the MoT parameters Online. The results are compared over a varying number of topics, up to 25.

The features used are dense SIFT [Lazebnik et al., 2006], as opposed to the gpu-sift used in the previous experiments. It is known that the selection of features can significantly affect the results, in [Fei-Fei and Perona, 2005b] a comparison is made between features, including dense SIFT, for natural scene categorisation. The results throughout the rest of this section are comparable. In [Sivic et al., 2005], a high classification rate is achieved using 4 classes from the CalTech dataset. They use a combination of MSER [Matas et al., 2004] and SIFT descriptors, making a direct comparison difficult.

Figure 7-9 shows the results of the experiment. Using the unsupervised

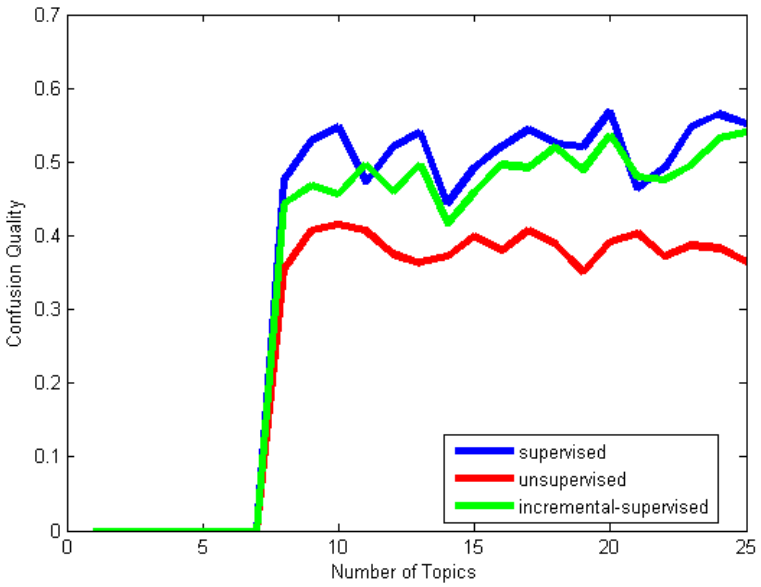


Figure 7-9: Confusion quality over a varying number of topics, on CalTech.

classifier provides a significantly lower quality of confusion than both supervised and online-supervised. If the test images were evenly distributed over the classes then, since there are 8 classes of objects, the confusion quality would average at 13%. Taking the random classifier as a benchmark, all of the graphs perform well. For optimal performance the supervised classifiers work best but require labelled data.

Tables 7.5, 7.6, and 7.7 show some sample confusion matrices, the highest quality of confusion is taken in each case. The 'car' and 'face' classes are consistently and accurately classified in each of the tests, implying that the data is well separated in the topic models. The number of images in each class of object is as follows {97, 122, 81, 114, 201, 800, 116, 435}, indicating that the number of training images does not correlate with how accurately the images are classified in to the correct class. The 'face' class has a large training set, and performs well; the aeroplane class has a large training set, but does not perform so well.

The results show a level of classification good enough to perform on a dataset with a high inter-class variance, with pictures of objects taken in a variety of different scenes, exhibiting change in scale and position. Although

0	Buddha	Guitar	Starfish	Umbrella	Watch	Aeroplane	Car	Face
Buddha	17	2	3	6	0	2	0	1
Guitar	5	5	1	10	2	2	1	5
Starfish	3	0	14	13	0	0	0	1
Umbrella	2	1	0	24	1	2	1	0
Watch	1	0	0	12	13	1	1	3
Airplane	0	0	0	12	0	14	5	0
Car	0	0	0	0	0	1	30	0
Face	2	0	0	2	3	0	0	24

Table 7.5: The optimal confusion matrix, built with a topic mixture of 20 topics. With CalTech images (supervised).

0	Buddha	Guitar	Starfish	Umbrella	Watch	Aeroplane	Car	Face
Buddha	13	8	3	3	2	1	0	1
Guitar	2	13	0	2	10	2	1	1
Starfish	5	5	18	2	1	0	0	0
Umbrella	2	12	1	10	5	0	1	0
Watch	4	4	2	3	16	2	0	0
Airplane	0	7	0	8	0	15	1	0
Car	0	0	0	0	0	5	26	0
Face	4	1	0	1	1	1	0	23

Table 7.6: The optimal confusion matrix, built with a topic mixture of 25 topics. With CalTech images (online-supervised).

0	Buddha	Guitar	Starfish	Umbrella	Watch	Aeroplane	Car	Face
Buddha	5	0	1	8	8	0	2	7
Guitar	1	0	2	10	10	1	4	3
Starfish	11	0	10	7	1	0	0	2
Umbrella	1	0	5	12	11	0	0	2
Watch	0	0	4	4	13	0	1	9
Airplane	1	0	4	8	3	7	8	0
Car	1	0	0	0	0	4	26	0
Face	0	0	0	0	1	0	0	30

Table 7.7: The optimal confusion matrix, built with a topic mixture of 10 topics. With CalTech images (unsupervised).

there is room for improvement, the algorithm could potentially be used for learning object classes on wide dataset of images such as Google image search, and used in a robotic system operating in the environment.

7.4 Discussion

This chapter presented a method for a robot to autonomously learn about objects in its environment using self-supervision. The models built by the robot can be used to recognise and classify the learned objects in novel scenes.

The extension of LDA in to a Mixture of Topics model, shows an improved quality of classification, but requires object level segmentation in order to determine the object from its background. This poses a problem for most robotic systems since, in order to obtain a good object level segmentation, some prior knowledge of the object is needed. In this chapter, the background is assumed to be plain, with fewer features than on the objects in the environment. The benefits of the using the more advanced segmentation algorithm introduced in Part II are explored in the next chapter.

Chapter 8

Combined Segmentation and Learning

The work discussed in the previous chapters of the thesis provide contributions to the literature in robotic perception and learning. The work presented in this chapter describes a method to combine perception with learning. The work integrates: *Object Segmentation*, using known robotic motion, allowing the robot to segment in the presence of moving and cluttered backgrounds; and *Learning*, building robust object models Online, without the need to recompute old data.

In summary, the work presented in this chapter supports the hypothesis in two ways,

- Motion is used in order to **segment** the objects the robot is interacting with. The algorithm is taken from Part II of the thesis, and is already proven to be robust to cluttered and changing environments. The robustness to cluttered environments is an improvement over the blob detection used in previous chapters, as it allows the robot to learn without the need to remove other objects from the camera field of view.

- A new **robotic planning module** is introduced to extend the self-supervision process. It uses a combination of learning, object recognition and motion planning to repeatedly push the object of interest, allowing it to be observed from multiple views.

The results show the algorithm to perform well in cluttered environments, achieving a classification accuracy of up to 90%. This section begins by giving

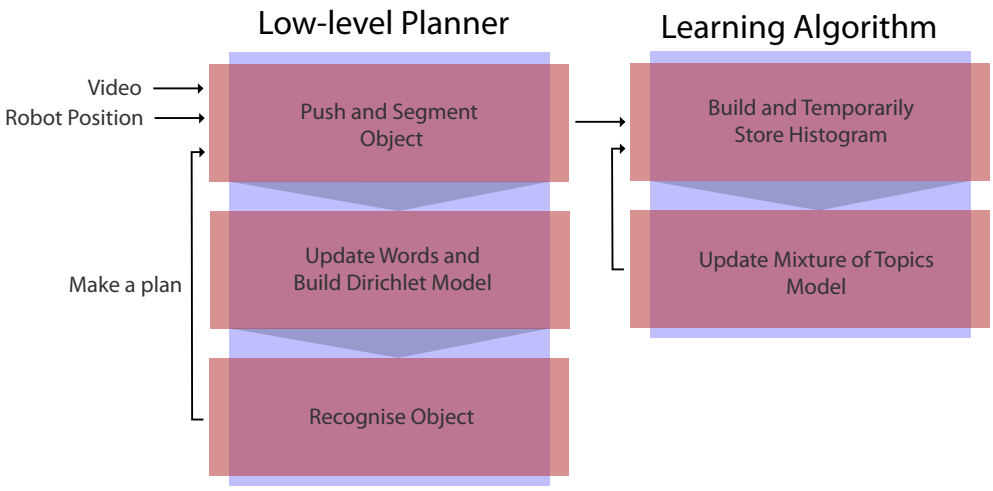


Figure 8-1: This diagram shows process of building a 'Mixture of Topics' model using the proposed system. Sensor data from the vision system and robot manipulator is passed to the algorithm, which iteratively plans a movement, segments an object, and recognises. This process generates a set of descriptors which are used to build a Mixture of Topics model for the objects in the environment.

an overview of the method, before describing the segmentation technique and planning algorithm in more detail.

8.1 System Overview

The goal of the system is to recursively push an object in the environment in order to build an object model, which can be used to classify the object again. The whole system is summarised in Figure 8-1, and is split in to two main subsystems. The first part is the planning algorithm, which segments objects and then finds the best way to induce motion on the objects; and the second part is the learning algorithm, i.e. the Mixture of Topics model used in previous chapters. The two subsystems are summarised below.

- **Low-level Planner**. In this module the robot makes a plan to push an object. When the object is moved, robotic bottom-up segmentation is used to determine the object from its background. The information is used to update a simple bag-of-words model, based on the Dirichlet distribution. The model is then used to recognise the object again, in

order to make a new plan to manipulate it. This process is iterated, passing segmented regions to the learning algorithm.

- **Learning Algorithm**. The data from the segmentation algorithm model is passed to a 'Mixture of Topics' model, shown to be able to classify objects with a high inter-class variance and with a robotic vision system (in Chapter 7).

The learning algorithm was detailed and tested in previous chapters. Throughout the rest of this section the low level planning algorithm is described in detail. The algorithm employs a 'look, plan, move' strategy. It begins by pushing the object and segmenting every frame, then using the segmentations to build a simple object model, finally, the model is used to recognise the object of interest again in the scene. Once the object has been found, the robot chooses a path to push the object through the centre of mass, without touching other objects in the environment. This process is repeated with each set of segmentations passed to the Mixture of Topics learning algorithm.

8.1.1 Bottom-Up Segmentation

The bottom-up segmentation algorithm is explained in detail in Part II, where *probabilistic motion models* are used to derive the probability that any pixel belongs to the object being manipulated. The approach used here is slightly different, in that SIFT features [Lowe, 2004a] are matched across frames to describe the motion, rather than optical flow. Also, instead of video segmentation, Mean Shift Clustering [Comaniciu and Meer, 2002] is used on the velocity vectors for each matched SIFT feature. Providing a low-level segmentation of the features at a particular frame. The motion models from Part II are then applied to the data, giving a probability distribution that a cluster of descriptors belong to the object. A threshold is then applied to the distribution to extract the object of interest.

The benefit of using a probabilistic approach to segmentation is that prior knowledge can be used to further improve the result, when an object model is available. The proceeding sections go on to describe how to build object models; or more specifically, find the probability that a feature descriptor belongs to a particular object.

8.1.2 Building a Bag-of-Words Model

The result of the above method is a set of features which belong to the object. This section details a simple method for building a bag-of-words model over the object features, using the Dirichlet distribution.

To build a bag-of-words model the first items needed are the 'words'. These correspond to clusters of all the features that have been observed, found by storing all of the features that have been observed and clustering them using Mean Shift [Comaniciu and Meer, 2002]. The centres of each distribution then become the words.

The set of words are calculated using the frames from each push. After the first push, Mean Shift is used on the descriptors belonging to the object. For all of the movements following the first, the words are incrementally updated. For each word ($\boldsymbol{w}$) the closest descriptors ($\boldsymbol{d}$) is found, replacing $\boldsymbol{w}$ with $\hat{\boldsymbol{w}}$ using the incremental mean update rule,

$$\hat{\boldsymbol{w}} = \frac{N}{N+1}(\boldsymbol{w} + \boldsymbol{d}). \tag{8.1}$$

Where N is the number of descriptors used to generate the word and $\hat{\boldsymbol{w}}$ is the new value for the word.

The goal is to then find the probability distribution of a word belonging to the object, given we are at frame i. Assuming a discrete set of words $\boldsymbol{w} \in W$ and a set of descriptions $\boldsymbol{d} \in D$ from a particular frame i. To find the distribution of words, marginalisation is used,

$$p(\boldsymbol{w}|i) = \sum_{\boldsymbol{d} \in D} p(\boldsymbol{w}|\boldsymbol{d}, i)p(\boldsymbol{d}|i) \tag{8.2}$$

$$= \sum_{\boldsymbol{d} \in D} N(\boldsymbol{w}|\boldsymbol{d}, \Sigma)p(\boldsymbol{d}|i). \tag{8.3}$$

This gives the distribution of the words at each frame; where, $p(\boldsymbol{d}|i)$ is the output of the segmentation at frame i, and Σ is the covariance associated with each word. There is now a set of distributions, for each frame, measuring the same object, these distributions are assumed to be Dirichlet distributed. To estimate the parameters of the Dirichlet distribution a combination of fixed-point iteration and Newton-Raphson as described in [Minka, 2003]. This

gives the object model used to recognise the object again after the robot has finished its movement.

It may be the case that the robot encounters extra noise when pushing an object, for instance, the arm partially covers the object. To avoid corrupting the model, it is updated incrementally. This is done by sampling the Dirichlet distribution using the parameters from a previous push, and concatenating it with the new data. The learning rate is set at 50:50, i.e. the number of samples generated by the old distribution is equal to the number of frames in the current push. The parameters of the Dirichlet distribution are then re-estimated using the new data.

8.1.3 Top-Down Recognition

After the object has been pushed, and a Bag-of-Words model learned over the data, the robot can use the model to recognise the object again. Performing this step means that if the object has moved from its previous position by an external force, the robot is able to adapt to the change.

To recognise the object, the parameters from the Dirichlet distribution described in Section 8.1.2 are used. The parameters of a Dirichlet distribution are themselves a probability distribution, and in the case described above, describe the probability that a word $\boldsymbol{w}$ belongs to the object being manipulated. This distribution is denoted $p(\boldsymbol{w})$.

When recognising, the algorithm has access to a set of descriptors and $p(\boldsymbol{w})$. The goal is to find the probability that a particular descriptor belongs to the object in question. To achieve this, marginalisation is used over the set of words,

$$p(\boldsymbol{d}) = \sum_{\boldsymbol{w} \in W} p(\boldsymbol{d}|\boldsymbol{w}) p(\boldsymbol{w}) \tag{8.4}$$

$$= \sum_{\boldsymbol{w} \in W} N(\boldsymbol{d}|\boldsymbol{w}, \hat{\Sigma}) p(\boldsymbol{w}). \tag{8.5}$$

Where the parameter $\hat{\Sigma}$ denotes the variance associated with each word. This can be trained when the dictionary is built, but in practice it makes little difference, and a small value (around 0.1) multiplied by the identity matrix suffices.

The distribution $p(\boldsymbol{d})$ can further be transformed into image space, to find the probability that a particular pixel $\boldsymbol{x}$ belongs to the object. The benefit of doing this is that it is a spatial distribution that also depends on the probability of a particular feature point belonging to the object. Analogously to Equation 8.5, marginalisation is performed over the descriptors,

$$p(\boldsymbol{x}) = \sum_{\boldsymbol{w} \in W} p(\boldsymbol{x}|\boldsymbol{d})p(\boldsymbol{d}) \tag{8.6}$$

The distribution $p(\boldsymbol{x}|\boldsymbol{d})$ is chosen as Gaussian, with a fixed covariance. To extract the object from the distributions a threshold is applied.

The threshold can be chosen by using the Dirichlet parameters associated with the object. The goal in doing this is to find a 'connected' subset of descriptions that have a high probability of being part of the object. The problem now is to find the threshold that has a high probability of containing the correct feature. Given the threshold κ, to find the subset of features which are contained in the foreground the following set is defined,

$$P(\kappa) = \{\boldsymbol{w} : p(\boldsymbol{w}) > \kappa\} \tag{8.7}$$

This set defines a region of the image containing the putative object. For each κ, the descriptors which are contained in the region build into a histogram over the set of words. The parameters of the Dirichlet distribution are then used on each histogram to find the threshold with the highest probability.

The result is a region of the image which is likely to contain the object interacted with by the robot. Once this object has been identified the robot can plan to move it. The algorithm used for planning is taken from Section 7.2.1, in Chapter 7. The components introduced above make up the low-level planning module, which is used to iteratively manipulate objects in the robot's workspace. The algorithm is tested for classification performance in the following section.

8.2 Experiments

The objective of these experiments are to show that the robot is able to gather data to build robust models of objects in the environment, using a combination of motion-based segmentation and learning. Videos are taken of the robot pushing 4 different objects, a 'book', 'bottle', 'CD' and toy 'giraffe', using the low-level planning algorithm described above. All of the videos have a cluttered background, with objects placed randomly around the object being manipulated. A set of sample frames of the robot pushing the toy giraffe is shown in Figure 8-2.

For each object there are three trials, with the robot pushing the objects an average of 10 times for each trial. The features used are standard SIFT features [Lowe, 2004b], but reimplemented to work on a GPU [Sinha et al., 2006]. This allows the algorithm to run quickly, a necessity for a robot to perform in a working environment.

The graphs shown in Figure 8-3a are the results of applying a supervised Mixture of Topics algorithm to the data, with a varying number of topics. The gray region shows the variance in changing the segmentation threshold, with values between 0.016 and 0.028. Increasing the number of topics does have an affect on the data, choosing a number of topics greater than 15 yields a higher quality of classification, on average, than lower values. Both graphs display a high variance, with confusion matrix accuracy as high as 90% and as low as 40%. Figure 8-3b shows the results of using an unsupervised Mixture of Topics model, the classification quality does not reach as high as for the supervised model, but is not significantly different.

Tables 8.1 and 8.2 show the best and worst confusion matrices, respectively. The matrices are the average over each threshold. Each of the rows show how the 20 different objects were classified, for instance, row 1 of Table 8.1 shows that the 20 images of a book given to the robot were correctly classified as a book 19 times, and incorrectly classified as a bottle once. Similarly the bottle was correctly classified 19 times out of 20, and incorrectly classified an average of 0.75 in to the CD class, and 0.25 in to the giraffe class.

Both confusion matrices show the giraffe to appear similar to the bottle and CD, and with the lowest classification rate among all of the objects. The book

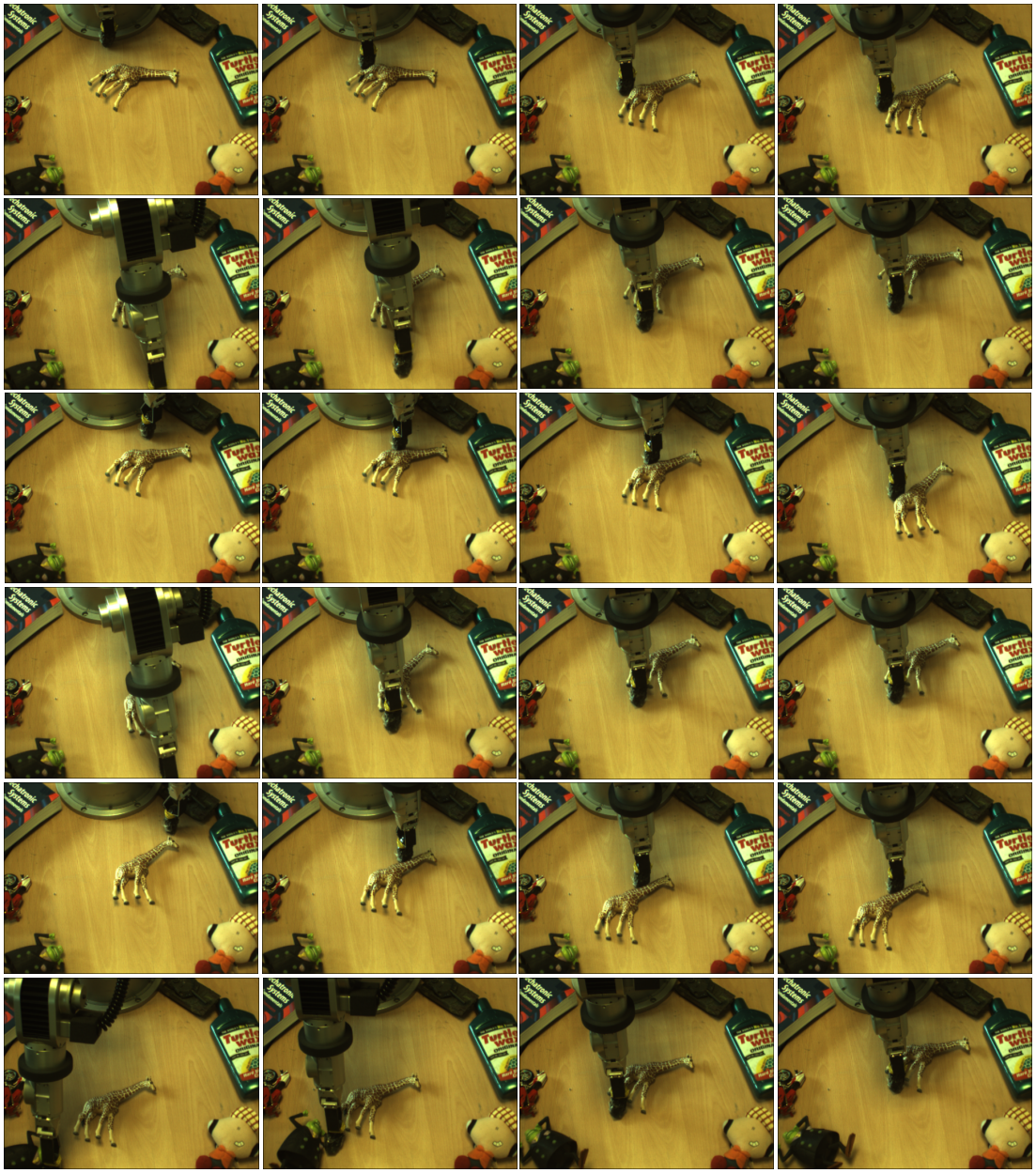
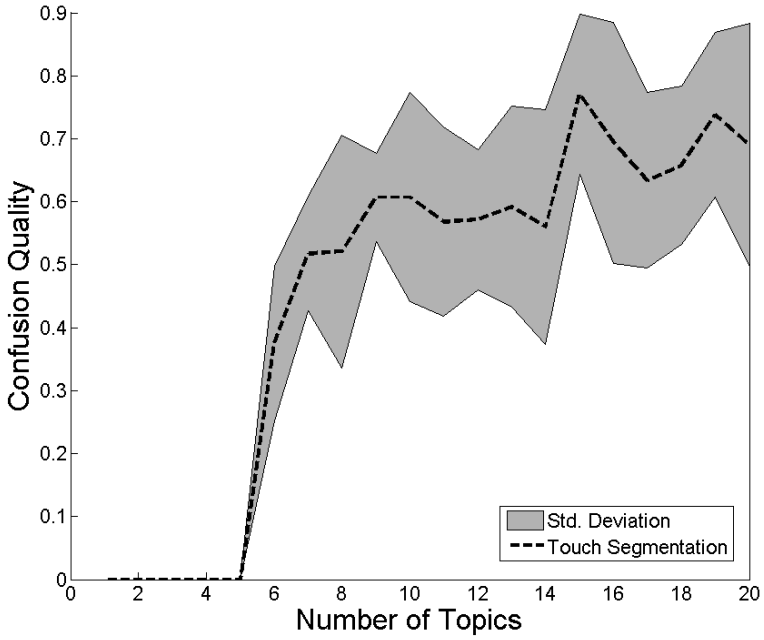
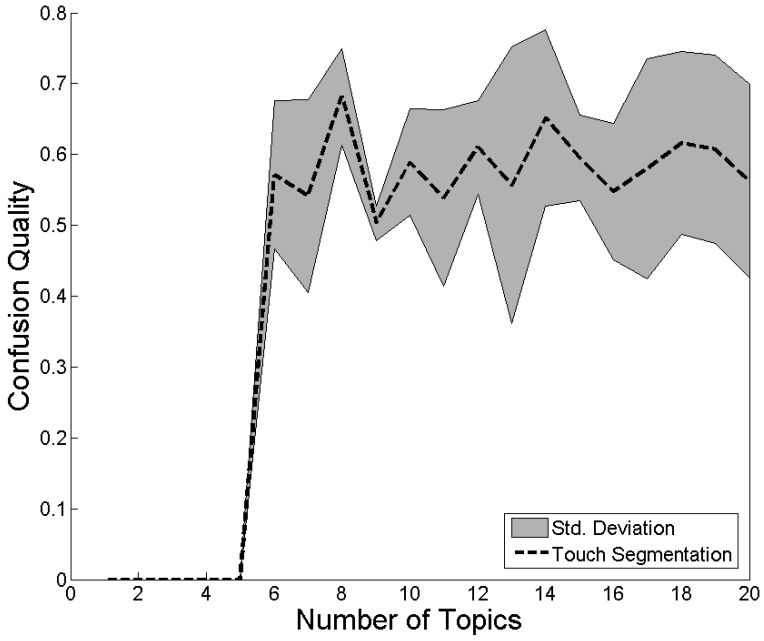


Figure 8-2: Sample frames of the robot manipulating the giraffe object. Each row represents a different push action. The robot continues with this strategy for 10 movements and 3 trials for each object. Note the background clutter and occlusion of the giraffe by the robot arm in several frames of the sequence.



(a) Supervised topic mixture using the robot.



(b) Unsupervised topic mixture using the robot.

Figure 8-3: Confusion accuracy in the Mixture of Topics model, with a varying number of topics. The variance in quality when choosing different segmentation thresholds is represented in grey, the dashed line represents the average quality.

is well classified in both of the matrices, since it is relatively visually distinct from the other objects due to the pattern on the front.

		Model			
		Book	Bottle	CD	Giraffe
Data	Book	19	1	0	0
	Bottle	0	19	0.75	0.25
	CD	0.5	3.5	17	1
	Giraffe	2.25	9.25	0.75	9.75

Table 8.1: Best average confusion matrix for 15 topics, over 4 different thresholds.

		Model			
		Book	Bottle	CD	Giraffe
Data	Book	17.5	2	0.5	0
	Bottle	0	0.5	19.5	0
	CD	0	1.25	19	1.75
	Giraffe	0	0.25	21.5	0.25

Table 8.2: Worst average confusion matrix for 6 topics, over 4 different thresholds.

Both of the experiments exhibit a large amount of noise, the quality reaches values as high as 90% but in some cases lower than 40%. Although the average quality of confusion is at a level high enough for practical use, the lower values may not be acceptable.

8.3 Discussion

This chapter presents an algorithm which is able to learn object models without human supervision. The algorithm requires a priori knowledge of the number of objects in the scene for unsupervised learning, the parameter K in the k-means algorithm. There is research available which can compute K by considering the information contained in each cluster [Sugar and James, 2003], but is not considered in this work.

Despite its limitations, the method presented in this chapter makes a significant effort towards an unsupervised means for a robot to learn about objects in the environment. The algorithm contributes a new planning

algorithm which is robust to cluttered an dynamic environments, following previous results in the thesis.

Part IV

Conclusions

Chapter 9

Conclusions

The work presented in this thesis provides a new method for interactively learning about objects using a robot. The method incorporates bottom-up object segmentation and top-down recognition into a system capable of classifying objects in the environment, robust to changes in viewpoint and with a large inter-class variance. The system is motivated by the idea that using induced robotic motion provides a benefit to the learning process. Each subsystem of the model makes explicit use of the robot sensor data to aid the vision system in learning object models.

The final system has an impact on industry and commerce, as it is capable of autonomously learning about objects in the environment. Any robot which needs to learn about the environment can be presented with a set of objects, making it more flexible for reuse with different applications. The user will not have to re-program the robot, or go to the effort of taking a set of pictures to retrain. The work also provides the first step towards a fully autonomous robot, which is capable of operating in the environment without any supervision. For instance, a robotic maid used to operate around a home would benefit from an algorithm capable of learning its own models, and the ability to update them with new information.

The contributions to the literature include a learning algorithm capable of operating in cluttered and dynamic environments. The model contributes online learning to the robotic community, allowing a robot to update centrally stored models without the the need to store or recompute old data.

This chapter concludes all of the results made throughout the thesis, and

discusses new areas for further research, which have been made possible as a result of the work. To begin with a discussion is made about the original hypothesis, and how the results support it. The main contributions of the thesis are presented, along with a chapter breakdown, discussing the benefits and limitations of each subsystem. Finally, a case for further work is presented, to extend and improve the overall results and method.

9.1 Hypothesis Support

Although there are a range of different contributions made to the robotics and computer vision literature, the hypothesis is as follows,

- **'Robot-object interaction, in the form of induced motion on objects, aids robot perception and the learning of visual object models.'**

Each chapter of the thesis provides some support for the hypothesis, and paves the way towards a system which utilises known robotic motion to aid the learning process:

- **Vision and Dynamics**: This chapter provides a case study into the use of dynamics and vision to intercept a moving target. Current readings are taken from the robot sensors, as a measure of the force exerted by each of the motors. Statistics of the vision system are combined with the dynamic information to build a probabilistic visual servo model. This does not directly support the hypothesis, since it does not involve learning or perception, but provides motivation that using extra sensor information to aid a vision problem can be beneficial. In the case of hand-eye coordination, it gives a means to simultaneously reduce the energy of a movement, while intercepting a moving target.

- **Robot Object Segmentation**: The work introduced in this chapter uses known robot motion to simultaneously segment and localise objects. Objects in the environment can be segmented using an algorithm which breaks a video into parts, using detected motion from the vision system. Robot segmentation is different because it not only segments the object,

but also localises it in the image. Without knowing the motion of the manipulator, this would not be possible. The work also shows that the known motion can accurately remove the robotic arm from the foreground, without a complex kinematic model, reducing the time it takes to calibrate the robot.

- **Building object models**: When learning object models, it is important that the objects are viewed from multiple angles, particularly since they need to be recognised. The work introduces a new model, based on the assumption that objects have more than one view, this assumption leads to more robust object models being built under viewpoint change. The standard method for learning in vision, is to give the system a set of pictures taken of objects in different scenes, with a set of different backgrounds. This is ideal if those images are available, for a robot learning to operate autonomously in a workplace, it may come across an object or class which has not been previously categorised by humans. In this case, the robot has to induce motion on the object to learn about its visual description from multiple views, and update its model.

- **Combined Segmentation and Learning**: This chapter combines robotic motion segmentation with the learning of object models, incorporating the benefits of both systems. The robot is able to learn in the presence of dynamic and cluttered environments, and also self-supervise the learning, allowing the robot to build object models from multiple views.

In summary, it is shown that motion can be used to segment and localise objects in cluttered and changing backgrounds. It is also shown that a Mixture of Topics model improves standard LDA when classifying objects. The MoT model needs object segmentation, and so the robot either needs to make assumptions about the background to remove it, and then physically remove other objects in the scene (self-supervision, as used in part III); or use robotic motion segmentation to remove the background, as used in chapter 8.

9.2 Limitations and Further Work

Although the thesis provides a significant effort towards the hypothesis and contributions such as online learning on a robotic system, there are gaps and limitations which would make good areas for further research.

The first and most important limitation worth noting is the online use of a Mixture of Topics model. Although the work uses Online LDA and MoT, each system works independently. This means that the robot has to play with the objects in two separate stages, once for the LDA model, and then again for the MoT model. Instead, the whole model can be formulated into one graphical model and computed online.

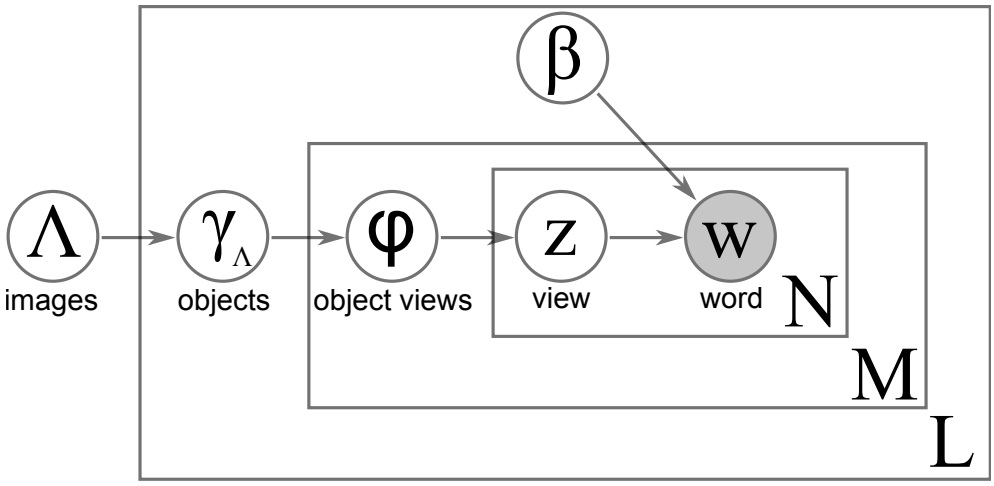


Figure 9-1: The 'Mixture of Topics' graphical model.

The 'Mixture of Topics' model is shown in Figure 9-1, and leads to the following factorisation,

$$p(\boldsymbol{w}, \boldsymbol{z}, \phi, \gamma_\Lambda, \beta|\Lambda) = \left[\prod_{n=1}^{N} p(w_n|z_n, \beta)p(z_n|\phi)\right] p(\phi|\gamma_\Lambda)p(\beta) \tag{9.1}$$

Integrating out ϕ and summing over z_n gives,

$$p(\boldsymbol{w}, \gamma_\Lambda, \beta|\Lambda) = p(\beta) \int \left[\prod_{n=1}^{N} \sum_{z_n} p(w_n|z_n, \beta) p(z_n|\phi) \right] p(\phi|\gamma_\Lambda) \partial\phi \tag{9.2}$$

This derivation is analogous to the derivation in [Blei et al., 2003] for Latent Dirichlet Allocation. Further following the working through and using variational Bayes, a fully online approach to Mixture of Topics for object recognition could be derived. The model is one hierarchy more complex than LDA, and finding a good factorisation for the variational probability distribution could be difficult. Performing the computation without needing to recompute or store any old data, would mean that a robot could continuously learn in the environment, with low memory requirements. The model must also be combined with an evolving (online) dictionary, which was covered to some extent in Chapter 6.3 via the use of a universal dictionary. As the robot encounters new data, new words can appear which are needed to describe the scene. Ideally the algorithm should require no knowledge or vocabulary, take batches of images as input, and return accurate class models.

The combination of segmentation and recognition used in Chapter 8 works well, but as mentioned in the discussion, requires the use of a short term memory module to manipulate the object. Using the long-term model to recognise the object while being manipulated would be more robust to the objects being moved between pushes.

9.3 Final Comments

The methods and results given in this thesis provide a method for autonomous learning using a robot. Despite the quality of the resulting system, and reliability in the face of noisy data, there is still a large amount of room for further research. The results show that the use of proprioception, a knowledge of relative body positions, on the robot can be used to improve the quality of unsupervised classification. The use of other sensor data such as torque readings, pressure readings from an artificial skin or sensors to measure heat transfer properties could be integrated in to the system further; improving the

performance of the learning algorithm.

Part V

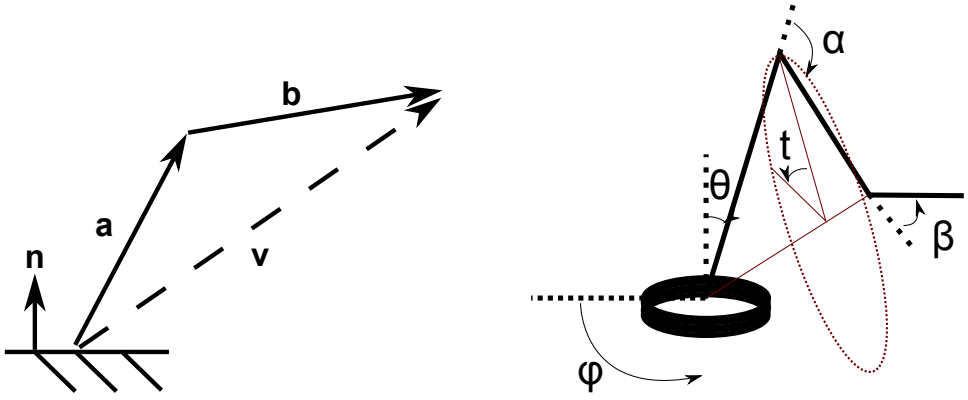
Appendix

Appendix A

Kinematics

A.1 Inverse Kinematics

In calculating the inverse kinematics of a robotic manipulator, the goal is to find the joint angles, given the position that the end-effector needs to be in. The result of this section is to solve the problem analytically. This is done by first solving the problem for a general 3 degree of freedom, 2 link robot; and then extending it to 4 degrees of freedom placing a geometric constraint on the final link.



(a) A diagram showing the geometry of a 2- link robot.

(b) The geometry of the final 4DOF robot.

Figure A-1: Diagrams of the robotic set-up. Both a simple 2 link robot, and a representation of the Katana manipulator used throughout the thesis.

The solution to the problem is as follows,

$$\alpha = sign(\mathbf{a}_{\perp}.\mathbf{b}) \cos^{-1}(\frac{\mathbf{a}.\mathbf{b}}{|\mathbf{a}||\mathbf{b}|}) \tag{A.1}$$

$$\theta = sign(\mathbf{o}.\mathbf{a}) \cos^{-1}(\frac{\mathbf{n}_3.\mathbf{a}}{|\mathbf{a}|}) \tag{A.2}$$

$$\phi = sign(\mathbf{n}_2.\mathbf{o}) \cos^{-1}(\frac{\mathbf{n}_1.\mathbf{o}}{|\mathbf{o}|}). \tag{A.3}$$

With the position of the end-effector is given by $\mathbf{v}$ let α denote the angle in the elbow joint (between $\mathbf{a}$ and $\mathbf{b}$) and let θ, ϕ denote the angles in the joint which connects the robot to the ground. Where θ is the angle between $\mathbf{a}$ and the ground. Assume that $\mathbf{n}_3$ is the unit normal. Let $\mathbf{n}_1, \mathbf{n}_2$ denote fixed normals perpendicular to $\mathbf{n}_3$. Where $\mathbf{o} = \frac{(\mathbf{n}_3 \times \mathbf{v}) \times \mathbf{n}_3}{|(\mathbf{n}_3 \times \mathbf{v}) \times \mathbf{n}_3|}$, is the outward xy normal.

To prove this result, vector geometry is used. Figure A-1 shows how the system of vectors is set up, with $\boldsymbol{a}$ and $\boldsymbol{b}$ the first and second links of the robot respectively.

$$\boldsymbol{v} = \boldsymbol{a} + \boldsymbol{b} \tag{A.4}$$

The problem is to find the joint angles given only $\mathbf{v}$, so $\mathbf{a}$ and $\mathbf{b}$ are not known. Assuming knowledge of $|\mathbf{a}|$ and $|\mathbf{b}|$ however, it is possible to find $\mathbf{a}.\mathbf{b}$.

$$|\mathbf{v}|^2 = \mathbf{v}.\mathbf{v} \tag{A.5}$$

$$= (\mathbf{a} + \mathbf{b}).(\mathbf{a} + \mathbf{b}) \tag{A.6}$$

$$= |\mathbf{a}|^2 + 2\mathbf{a}.\mathbf{b} + |\mathbf{b}|^2 \tag{A.7}$$

Hence,

$$\mathbf{a}.\mathbf{b} = \frac{1}{2}(|\mathbf{v}|^2 - (|\mathbf{a}|^2 + |\mathbf{b}|^2)) \tag{A.8}$$

Using equation (1) it follows that,

$$\mathbf{a}.\mathbf{v} = |\mathbf{a}|^2 + \mathbf{a}.\mathbf{b} \tag{A.9}$$

Since $|\mathbf{a}|^2 + \mathbf{a}.\mathbf{b}$ is constant. It follows that $\mathbf{a}$ lies on the plane with normal $\mathbf{n} := \frac{\mathbf{v}}{|\mathbf{v}|}$ and distance from the origin $c := \frac{1}{|\mathbf{v}|}(|\mathbf{a}|^2 + \mathbf{a}.\mathbf{b})$. To find the possible

positions the following simultaneous equations need to be solved,

$$\mathbf{x}.\mathbf{n} = c \tag{A.10}$$

$$|\mathbf{x}| = |\mathbf{a}| \tag{A.11}$$

For each $\mathbf{x} \in \mathbb{R}$. Conceptually it is clear that this is the intersection of a plane and the boundry of a ball. So this is a nonlinear problem, with no solutions, one solution or infinitely many solutions (lying on a circle) depending on the value of c. Ideally we want to control the position of the elbow, so a parametrisation of the circle is needed.

Consider the transformed problem defined by the equations,

$$\mathbf{x}.\widetilde{\mathbf{n}} = 0 \tag{A.12}$$

$$|\mathbf{x} - \widetilde{\mathbf{n}}c| = |\mathbf{a}| \tag{A.13}$$

Where $\widetilde{\mathbf{n}}$ is normal to the plane $\mathbf{x}_3 = 0$. It follows that,

$$x_1^2 + x_2^2 = |\mathbf{a}|^2 - c^2 \tag{A.14}$$

This gives the radius of the circle to be parametrised. Which is then given by,

$$\mathbf{a}(t) = \frac{c\mathbf{v}}{|\mathbf{v}|} + (|\mathbf{a}|^2 - c^2)^{\frac{1}{2}}(\sin(t)\tilde{\mathbf{n}}_1 + \cos(t)\tilde{\mathbf{n}}_2) \tag{A.15}$$

Where,

$$\tilde{\mathbf{n}}_1 = \frac{\mathbf{v} \times \mathbf{n}_3}{|\mathbf{v} \times \mathbf{n}_3|} \tag{A.16}$$

$$\tilde{\mathbf{n}}_2 = \frac{(\mathbf{v} \times \mathbf{n}_3) \times \mathbf{n}}{|(\mathbf{v} \times \mathbf{n}_3) \times \mathbf{n}|} \tag{A.17}$$

This provides all the necessary information to calculate the joint angles for a 2 link robot, given in Equations A.1, A.2, A.3. To find the joint angles for a 4DOF robot, as shown in Figure A-2, a constraint is chosen, specifically the angle $\boldsymbol{c}$ makes with the horizontal. The correct position for link $\boldsymbol{b}$ is then calculated using the method defined above. Here we know $|\mathbf{a}|, |\mathbf{b}|, |\mathbf{c}|$ and the position vector $\mathbf{v}$. .

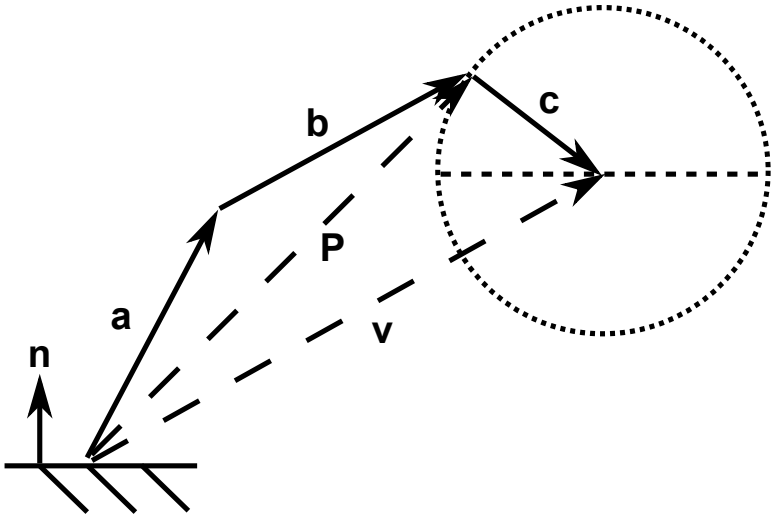


Figure A-2: Diagram of a 4DOF, 3 link robot.

Letting ψ denote the angle the gripper will be accessing the object from. Then,

$$\mathbf{P} = \mathbf{v} + |\mathbf{c}|(\sin(\psi)\mathbf{n}_3 + \cos(\psi)\mathbf{o}) \tag{A.18}$$

The angles α, θ and ϕ. Can then be found using the method defined above, but since there are only three degrees of freedom in the remaining problem an extra-constraint is needed, to ensure that $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ are coplanar. Specifically,

$$\mathbf{x}(t).\frac{\mathbf{a} \times \mathbf{b}}{|\mathbf{a} \times \mathbf{b}|} = 0. \tag{A.19}$$

This happens when $t \in \{0, \pi\}$. In practise, t is chosen to be π, ensuring the robots elbow is convex, as drawn in Figure A-2.

Appendix B

Learning

B.1 Bags of Words for Text Analysis

The 'bag of words' model dates back to the 60's [Maron and Kuhns, 1960] for analysing text documents. The model is based on the simplifying assumption that words in a document are unordered, and that a document can be represented by the frequency of the words within the document alone. Mathematically speaking, we assume that the each word is statistically independent of other words in the document.

This idea relates to the problem in computer vision of object classification. If an object can be represented by a set of low level features (words). Then a bag of words model represents the object as the frequency of features contained within it, eg. a car has two wheels and four doors, where as a bike has two wheels and zeros doors. Analogously to the document classification problem, the spatial location of the features are not taken in to account.

B.1.1 Supervised Learning

If we have a set of training data in whiich the classes are already identified, building a model to represent the data is known as *supervised* learning. For example, a set of articles along with their classes (Computer Vision, Robotics, Machine Learning) are given to the algorithm, which should then be able to accurately classify a new (previously unseen) document in to one of the classes.

There are several supervised algorithms known to the machine learning

literature. This section covers the most basic bag of words model, known as *naive bayes*. Suppose we have a set of features $f_i \in \mathbb{R}^N$ and a set of classes $C \in \mathbb{N}$. The problem is to find the probability of a class, given the set of features. Bayes theorem obtains,

$$p(C|f_1, f_2, ..., f_K) \propto p(f_1, f_2, ..., f_K|C)p(C). \tag{B.1}$$

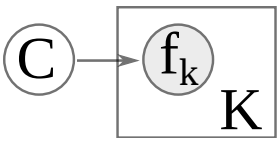


Figure B-1: The naive Bayes model.

The words are then assumed statistically independent and so Equation B.1 factorises in to,

$$p(C|f_1, f_2, ..., f_K) \propto p(C) \prod_{k=1}^{K} p(f_k|C). \tag{B.2}$$

Finding the maximum of Equation B.2 yeilds the class C most likely to belong to the given words. The $p(f_k|C)$ likelihoods must be trained on a set of examples beforehand. There are several ways to train this likelihood function given a set of training images.A non-parametric way is to use density estimation, which can be slow. The usual approach is to use kernel density estimation (KDE) [Scott, 1992]. Given a set of L samples $\boldsymbol{d}_1, ..., \boldsymbol{d}_L \in \mathbb{R}^N$ from class C the probability density function $p(\cdot|C)$ is estimated as follows,

$$p(\boldsymbol{f}|C) = \frac{1}{L} \sum_{n=1}^{L} K_f(\boldsymbol{f} - \boldsymbol{d}_n). \tag{B.3}$$

The function K_f is a kernel function, usually a Gaussian or flat kernel. The samples are the concatination of all training image descriptors. Using a parameteric approach can less time consuming, memory intensive, and depending on the method, made to work incrementally. These methods make extra assumptions about the distribution of the features in an image. An example of a parameteric estimation is to assume the features form a mixture of

Gaussians distribution, the parameters of the model can be estimated using the EM-algorithm.

In the more specific case of text analysis, the features are words, drawn from a discrete distribution. Rather than a set of continuous features, f_k, some words will be equivalent in a document. The distribution $p(f_k|C)$ in equation B.2, has a discrete support. Letting $\boldsymbol{h} \in \mathbb{N}^N$ denote a histogram, i.e. h_n is the number of words in the document that correspond with dictionary word n. Some of the words in Equation B.2 then become equivalent, denoting the words in the dictionary $\boldsymbol{\omega}$. We have,

$$p(\boldsymbol{\omega}|\boldsymbol{h}, C) = \prod_{k=1}^{K} p(\omega_k|\boldsymbol{h}, C)^{h_k} \tag{B.4}$$

$$= \prod_{k=1}^{K} p(\omega_k|C)^{h_k} \tag{B.5}$$

Using conditional bayes we have,

$$p(\boldsymbol{h}|\boldsymbol{\omega}, C) = \prod_{k=1}^{K} p(\omega_k|C)^{h_k} \frac{p(\boldsymbol{h})}{p(\boldsymbol{\omega})} \tag{B.6}$$

Letting $\theta_k = p(\omega_k|C)$, we can rewrite the Equation as follows,

$$p(\boldsymbol{h}|\boldsymbol{\theta}) = \prod_{k=1}^{K} \theta_k^{h_k} \frac{p(\boldsymbol{h})}{p(\boldsymbol{\omega})} \tag{B.7}$$

Assuming a flat distribution over the words, and normalising gives the multinomial distribution,

$$p(\boldsymbol{h}|\boldsymbol{\theta}) = \frac{K!}{\prod_{i=1}^{K} h_i!} \prod_{i=1}^{K} \theta_i^{h_i} \tag{B.8}$$

A class of documents is then represented by the parameters $\boldsymbol{\theta}$. Bayes theorem is then used to find the probability of a class given a histogram over the words in a document.

$$p(\boldsymbol{\theta}|\boldsymbol{h}) \propto p(\boldsymbol{h}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) \tag{B.9}$$

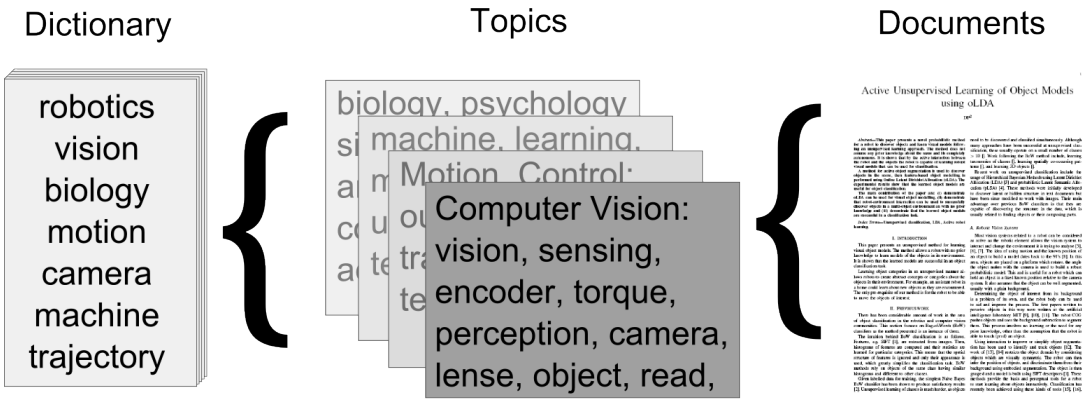
This version of naive Bayes , specific to discrete distributions is known as a *Mixture of Unigrams* model.

B.1.2 Unsupervised Topic Modelling

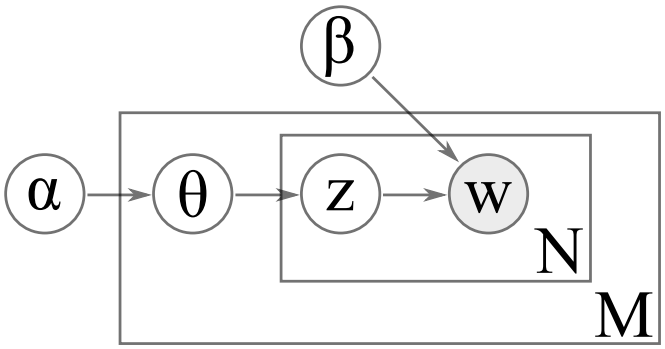
Supervised classification algorithms are useful when labelled training data is available to the system. This isn't always the case, and it is more generally desireable to give the program a set of features and allow it to distinguish the seperate classes automatically. Bag of words methods exist to separate data in this way, the most prominent in literature are Probabilistic Latent Semantic Analysis (pLSA) and Latent Dirichlet Allocation(LDA). Both of these methods are *generative*, meaning that the distribution of the data is modelled according to the way it was generated. The complement of these methods are known as *discriminative* models, where the means by which the data is generated is not considered. Discrimintive models tend to be faster and are applicable to a wider range of data sets, generative models are specific but more accurate.

Latent Dirichlet Allocation (LDA) was originally presented as a probabilistic generative model to discover the latent 'topics' in text documents [Blei et al., 2003]. The idea is that each document comprises a random mixture of topics, and that each topic specifies a distribution that governs a collection of words drawn at random from a vocabulary. LDA is an unsupervised approach to classification, the dictionary of visual words is given, along with a number of documents. The problem is to find the parameters of the multinomial distributions that make up the objects.

One of the main problems of learning algorithms is that they need the whole training set to build the model. If new data becomes available to learn from the algorithm needs to recompute the model for the old and new data again. This means that all of the data would need to be stored along with the model, which is memory intensive. Also, recomputing models over large training sets can take a long time, if new data is constantly made available the computation time needed for the update can become intractable. To get around this problem, algorithms are build which don't need to recompute over all of the data, these are known as *incremental* learning algorithms. Further, if the algorithm can be made to update its model without needing to *store* all of the data, it is known



(a) Hierarchical representation as used in LDA



(b) LDA Graphical Model

Figure B-2: The LDA data structure.

as an *online* learning algorithm. This section describes the Latent Dirichlet Allocation method for learning, and the steps needed to make it operate Online.

Latent Dirichlet Allocation

Similarly to the mixture of unigrams model in Section B.1.1, the images are represented as histograms over visual words. The histogram for each object is assumed to be drawn from a multinomial distribution, but in this case we don't assume that each document contains a single object, but a proportion of the features from each object.

The original paper on LDA [Blei et al., 2003] describes the model as a generative process. We have M images containing a distribution of objects θ, the distribution of objects are dirichlet distributed, conditioned by a parameter

α,

$$p(\theta|\alpha) = Dir(\theta|\alpha). \tag{B.10}$$

The distribution of objects represents the image, from this distribution a particular object $z \in \mathbb{N}$ can be drawn. Once we have an object, the histogram of words contained inside the object are assumed to be multinomially distributed, conditioned on a parameter $\boldsymbol{\beta}_z$,

$$p(\boldsymbol{h}|\beta_z) = Mult(\boldsymbol{h}|\boldsymbol{\beta}_z) \tag{B.11}$$

This allows a single word to be drawn from the histogram for each of the N words in the object. The parameter β is a table, with the number of rows and columns equivalent to the total number of objects and words respectively. The $\boldsymbol{\beta}_z$ make up the rows of β.

Figure B-2a illustrates a dictionary of available words on the left, topics in the centre and documents in the right. The words that appear in any given document are drawn from dictionary following the distributions that the topics dictate. This representation is usually known as hierarchical, as topics are abstractions over words that dictate their structure or distribution.

Figure B-2b shows the graphical model of this generative process using plate notation. The words $\boldsymbol{w}$ are the only observed variables, all of the other parameters must be computed. In this case *variational inference* is used, described in more detail in [Blei et al., 2003].

The graphical model shows the independence of variables, which is equivalent to the following probability distribution,

$$p(\theta, \boldsymbol{z}, \boldsymbol{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta) \tag{B.12}$$

Given the relationships just described there are only two free parameters, α and η. The likelihood of all other parameters, including the observed data w, is

given by

$$p(w, z, \theta, \beta|\alpha, \eta) = \prod_{i=1}^{K} p(\beta_i|\eta) \prod_{j=1}^{M} p(\theta_j|\alpha) \prod_{k=1}^{N} p(z_{j,k}|\theta_j) p(w_{j,k}|\beta_{z_{j,k}}). \quad \text{(B.13)}$$

In which K is the size of the vocabulary, M is the number of documents, and N is the number of words (this should be written N_j to show its dependence on the choice of document). The key problem to solve for LDA is computing the posterior of the latent variables, θ, z, and β; that is

$$p(\theta, z, \beta|w, \alpha, \eta) = \frac{p(\theta, z, w, \beta|\alpha, \eta)}{p(w|\alpha, \eta)}. \quad \text{(B.14)}$$

Unfortunately, estimating the normalising term

$$p(w|\alpha, \eta) = \int \int \sum_{z} p(w, z, \theta, \beta|\alpha, \eta) \quad \text{(B.15)}$$

is intractable in general. Solutions are found using either Monte Carlo Markov Chain or variational Bayes. The variational approach to estimating $p(w|\alpha, \eta)$ is used for online LDA, so its salient points are outlined here — but see [Blei et al., 2003] for details.

The variational Bayes approach to estimating $p(w|\alpha, \eta)$ uses a simpler distribution — one that can be factorised. The probability that the i^{th} word of the vocabulary appears in document j, which include topic k, is $q(w_{jik}) = q(z_{jk}, \theta_j, \beta_i)$; which factorises into $q(z_{ji} = k) = \phi_{jik}$; $q(\theta_j) = \text{Dir}(\theta_j; \gamma_j)$; $q(\beta_i) = \text{Dir}(\beta_i; \lambda_i)$. The variational parameters, (γ, ϕ, λ), are evaluated by minimising the Kullback-Leibler divergence between the variational distribution and the true posterior, thus:

$$(\gamma^*, \phi^*, \lambda^*) = \underset{(\gamma,\phi,\lambda)}{\text{argmax}}\ KL(q(\theta, z, \beta|\gamma, \phi, \lambda)||p(\theta, z, \beta|w, \alpha, \eta)).$$

An EM approach is used to find a solution. The E-step finds optimising values of the variational parameters γ, ϕ for all documents d in a corpus D. The

M-step updates λ, and also the model parameters α and η. In the **E-step** optimal values for γ and ϕ are found by gradient descent. That is, iterate over

$$\phi_{jik} \propto \exp\{E_q[\ln\theta_{jk}] + E_q[\ln\beta_{ki}]\}, \tag{B.16}$$

$$\gamma_{jk} = \alpha + \sum_i n_{ji}\phi_{jik} \tag{B.17}$$

until the average change in γ_{jk} is pleasingly small (about 10^{-5}); n_{ji} counts the number of times word i appears in document j. The expectations are then

$$E_q[\ln\theta_{jk}] = \Psi(\gamma_{jk}) - \Psi(\sum_{w=1}^{N}\gamma_{jw}) \tag{B.18}$$

$$E_q[\ln\beta_{ki}] = \Psi(\lambda_{ki}) - \Psi(\sum_{w=1}^{K}\lambda_{kw}) \tag{B.19}$$

in which Ψ is the first derivative of the logarithm of the gamma function (i.e. the digamma function).

The **M-step** involves updating the λ, β, α and η functions. The variable λ is updated as follows,

$$\lambda_{ki} = \eta + \sum_{j=1}^{N}\phi_{jik}n_{ji}. \tag{B.20}$$

Where n_{ji} is the number of times that word i appears in document j. The α and η parameters are updated using a Newton-Raphson procedure

$$\alpha \leftarrow \alpha - \rho_t\bar{\alpha}(\gamma_t), \tag{B.21}$$

$$\eta \leftarrow \eta - \rho_t\bar{\eta}(\lambda). \tag{B.22}$$

In this the $\bar{\alpha}$ and $\bar{\eta}$ are the step displacements, each the product of an inverse Hessian with a gradient vector; see [Blei et al., 2003].

Online LDA

The online version of LDA is almost identical to the batch version just outlined, see [Blei et al., 2003] for a more in-depth explanation than can be given here.

The technique uses "mini-batches", that is the new data is processed in chunks, rather that one at a time; this mitigates noise. Only the M-step changes. In particular the "topics" are estimated as a linear combination of the new value just computed with previous values. If a corpus of D documents is available for training, and a new batch of size S arrives:

$$\bar{\lambda}_{kw} = \eta + \frac{D}{S} \sum_{s} n_{tsw} \phi_{tswk}, \tag{B.23}$$

$$\lambda = (1 - \rho_t)\lambda + \rho_t \bar{\lambda}. \tag{B.24}$$

The mixture proportion, ρ_t, controls the influence of the new batch of data compared with the current model. It is set so that new data has lower influence as time progresses; in particular $\rho_t = (\tau_0 + t)^{\kappa}$, for $\kappa \in (0.5, 1]$ to ensure convergence. A limitation is that D should be known in advance.

References:

[Agarwal and Roth, 2002] Agarwal, S. and Roth, D. (2002). Learning a sparse representation for object detecttion. In *European Conference on Computer Vision*, pages 113–130.

[Alex Flint, 2011] Alex Flint, David Murray, I. R. (2011). Manhattan scene understanding using monocular, stereo, and 3d features. In *International Conference on Computer Vision (ICCV)*.

[Allen et al., 1993] Allen, P., Timcenko, A., Yoshimi, B., and Michelman, P. (1993). Automated tracking and grasping of a moving object with a robotic hand-eye system. *Robotics and Automation, IEEE Transactions on*, 9(2):152–165.

[Amit and Geman, 1999] Amit, Y. and Geman, D. (1999). A computational model for visual selection. *Neural Computation*, 11(7):1691–1715.

[Apostoloff and Fitzgibbon, 2006] Apostoloff, N. and Fitzgibbon, A. (2006). Automatic video segmentation using spatiotemporal t-junctions. In *Proc. BMVC*.

[Arbeláez et al., 2009] Arbeláez, P., Maire, M., Fowlkes, C., and Malik, J. (2009). From contours to regions: An empirical evaluation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2294–2301. IEEE.

[Arsenio et al., 2003] Arsenio, A., Fitzpatrick, P., Kemp, C., and Metta, G. (2003). The whole world in your hand: Active and interactive segmentation. In *Citeseer*. Citeseer.

[Asada et al., 2009] Asada, M., Hosoda, K., Kuniyoshi, Y., Ishiguro, H., Inui, T., Yoshikawa, Y., Ogino, M., and Yoshida, C. (2009). Cognitive developmental robotics: a survey. *Autonomous Mental Development, IEEE Transactions on*, 1(1):12–34.

[Baeten and Schutter, 2004] Baeten, J. and Schutter, J. (2004). *Integrated visual servoing and force control: the task frame approach*, volume 8. Springer Verlag.

[Banerjee and Basu, 2007] Banerjee, A. and Basu, S. (2007). Topic models over text streams: a study of batch and onlineunsupervised learning. In *SIAM Intl. Conf. on Data Mining*.

[Barber et al., 1996] Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls. *ACM TRANSACTIONS ON MATHEMATICAL SOFTWARE*, 22(4):469–483.

[Beale et al., 2010] Beale, D., Iravani, P., and Hall, P. (2010). Statistical visual-dynamic model for hand-eye coordination. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 3931–3936. IEEE.

[Beale et al., 2011a] Beale, D., Iravani, P., and Hall, P. (2011a). Probabilistic models for robot-based object segmentation. *Robotics and Autonomous Systems*.

[Beale et al., 2011b] Beale, D., Iravani, P., Hall, P., Charron, C., and Hicks, Y. (2011b). Visual object classification by robots, using on-line, self-supervised learning. In *1st IEEE Workshop on Challenges and Opportunities in Robot Perception, ICCV*.

[Biederman, 1987] Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. *Psychological review*, 94(2):115.

[Bilmes, 1998] Bilmes, J. (1998). A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models. *International Computer Science Institute*, 4:126.

[Bishop, 2006] Bishop, C. (2006). *Pattern recognition and machine learning*, volume 4. springer New York.

[Blei et al., 2003] Blei, D., Ng, A., and Jordan, M. (2003). Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1022.

[Borotschnig et al., 1999] Borotschnig, H., Paletta, L., and Pinz, A. (1999). A comparison of probabilistic, possibilistic and evidence theoretic fusion schemes for active object recognition. *Computing*, 62(4):293–319.

[Bosch et al., 2007] Bosch, A., Zisserman, A., and Muñoz, X. (2007). Image classification using random forests and ferns. In *International Conference on Computer Vision*.

[Bouguet, 2001] Bouguet, J. (2001). Pyramidal implementation of the affine lucas kanade feature tracker:description of the algorithm. Technical report, Technical report). Intel Corporation.

[Boykov and Kolmogorov, 2004] Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137.

[Boykov et al., 2001] Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239.

[Bradski and Kaehler, 2008] Bradski, G. and Kaehler, A. (2008). *Learning opencv*. O'Reilly.

[Brooks, 1999] Brooks, R. (1999). *Cambrian intelligence: the early history of the new AI*. The MIT press.

[Brox and Malik, 2010] Brox, T. and Malik, J. (2010). Object segmentation by long term analysis of point trajectories. In *Computer Vision–ECCV 2010*, pages 282–295. Springer.

[Burl et al., 1998] Burl, M., Weber, M., and Perona, P. (1998). A probabilistic approach to object recognition using local photometry and global geometry. In *European Conference on Computer Vision*, pages 628–641.

[Canini et al., 2009] Canini, K., Shi, L., and Griffiths, T. (2009). Online inference of topics with latent dirichlet allocation. In *International Conference on Articial Intelligence and Statistics*, volume 5.

[Cao and Fei-Fei, 2007] Cao, L. and Fei-Fei, L. (2007). Spatially coherent latent topic model for concurrent object segmentation and classification. In *Proceedings of IEEE Intern. Conf. in Computer Vision (ICCV).*

[Capparella et al., 2005] Capparella, F., Freda, L., Malagnino, M., and Oriolo, G. (2005). Visual servoing of a wheeled mobile robot for intercepting a moving object. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2737–2743. IEEE.

[Chaumette and Hutchinson, 2006] Chaumette, F. and Hutchinson, S. (2006). Visual servo control. i. basic approaches. *Robotics & Automation Magazine, IEEE*, 13(4):82–90.

[Chaumette and Hutchinson, 2007] Chaumette, F. and Hutchinson, S. (2007). Visual servo control. ii. advanced approaches [tutorial]. *Robotics & Automation Magazine, IEEE*, 14(1):109–118.

[Chen, 2008] Chen, S. (2008). *Active sensor planning for multiview vision tasks*. Springer Verlag.

[Cheng, 1995] Cheng, Y. (1995). Mean shift, mode seeking, and clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 17(8):790–799.

[Clarkson, 2006] Clarkson, K. (2006). Nearest-neighbor searching and metric space dimensions. *Nearest-Neighbor Methods for Learning and Vision: Theory and Practice*, pages 15–59.

[Coleman et al., 1992] Coleman, T., Li, Y., Center, C. T., and Institute, C. T. C. A. C. R. (1992). *On the convergence of reflective Newton methods for large-scale nonlinear minimization subject to bounds*. Citeseer.

[Comaniciu and Meer, 2002] Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24:603–619.

[Comaniciu et al., 2001] Comaniciu, D., Ramesh, V., and Meer, P. (2001). The variable bandwidth mean shift and data-driven scale selection. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 438–445. IEEE.

[Conte and Doherty, 2008] Conte, G. and Doherty, P. (2008). An integrated uav navigation system based on aerial image matching. In *Aerospace Conference, 2008 IEEE*, pages 1–10. Ieee.

[Corke and Hutchinson, 2001] Corke, P. and Hutchinson, S. (2001). A new partitioned approach to image-based visual servo control. *Robotics and Automation, IEEE Transactions on*, 17(4):507–515.

[Croon and Postma, 2006] Croon, G. and Postma, E. (2006). Comparing active vision models. *Image and Vision Computing*, 27:4.

[Davidson, 1998] Davidson, A. J. (1998). *Mobile Robot Navigation Using Active Vision*. PhD thesis, University of Oxford.

[Fei-Fei and Perona, 2005a] Fei-Fei, L. and Perona, P. (2005a). A bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition*.

[Fei-Fei and Perona, 2005b] Fei-Fei, L. and Perona, P. (2005b). A bayesian hierarchical model for learning natural scene categories. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 524–531. Ieee.

[Felzenszwalb and Huttenlocher, 2004] Felzenszwalb, P. and Huttenlocher, D. (2004). Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181.

[Fergus et al., 2010] Fergus, R., Bernal, H., Weiss, Y., and Torralba, A. (2010). Semantic label sharing for learning with many categories. In Daniilidis, K., Maragos, P., and Paragios, N., editors, *Computer Vision – ECCV 2010*, volume 6311 of *Lecture Notes in Computer Science*, pages 762–775. Springer.

[Fergus et al., 2003] Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Computer Vision and Pattern Recognition*.

[Ferrari et al., 2010] Ferrari, V., Jurie, F., and Schmid, C. (2010). From images to shape models for object detection. *International Journal of Computer Vision*.

[Fitzpatrick, 2002] Fitzpatrick, P. (2002). First contact: an active vision approach to segmentation. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 3, pages 2161–2166. IEEE.

[Fitzpatrick and Metta, 2003] Fitzpatrick, P. and Metta, G. (2003). Grounding vision through experimental manipulation. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 361(1811):2165.

[Fitzpatrick et al., 2008] Fitzpatrick, P., Needham, A., Natale, L., and Metta, G. (2008). Shared challenges in object perception for robots and infants. *Infant and Child Development*, 17(1):7–24.

[Forssen and Lowe, 2007] Forssen, P. and Lowe, D. (2007). Shape descriptors for maximally stable extremal regions. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. Ieee.

[Fu et al., 2010] Fu, Z., Lu, H., and Li, W. (2010). Incremental visual objects clustering with the growing vocabulary tree. *Multimedia Tools and Applications*, pages 1–18.

[Gibbens et al., 2000] Gibbens, P., Dissanayake, G., and Durrant-Whyte, H. (2000). A closed form solution to the single degree of freedom simultaneous localisation and map building (slam) problem. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, volume 1, pages 191–196. IEEE.

[Gibson, 1986] Gibson, J. (1986). *The ecological approach to visual perception*. Lawrence Erlbaum.

[Griffin et al., 2007] Griffin, G., Holub, A., and Perona, P. (2007). Caltech-256 object category dataset.

[Griffin and Perona, 2008] Griffin, G. and Perona, P. (2008). Learning and using taxonomies for fast visual categorization. In *Computer Vision and Pattern Recognition*.

[Griffiths and Steyvers, 2004] Griffiths, T. and Steyvers, M. (2004). Finding scientific topics. In *National Academy of Sciences of the USA*, volume 101, pages 5228 – 5235.

[Grundmann et al., 2010] Grundmann, M., Kwatra, V., Han, M., and Essa, I. (2010). Efficient hierarchical graph-based video segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2141–2148. IEEE.

[Guerrero et al., 2008] Guerrero, P., Ruiz-del Solar, J., and Díaz, G. (2008). Probabilistic decision making in robot soccer. *RoboCup 2007: Robot Soccer World Cup XI*, pages 29–40.

[Hartley and Zisserman, 2000] Hartley, R. and Zisserman, A. (2000). *Multiple View Geometry - Second Edition*, volume 6. Cambridge University Press.

[Hoffman et al., 2010a] Hoffman, M., Blei, D., and Bach, F. (2010a). Online learning for latent dirichlet allocation. *Advances in Neural Information Processing Systems*, 23:856–864.

[Hoffman et al., 2010b] Hoffman, M., Blei, D., and Bach, F. (2010b). Online learning for latent dirichlet allocation. In *NIPS*.

[Horn and Schunck, 1981] Horn, B. and Schunck, B. (1981). Determining optical flow. *Artificial intelligence*, 17(1-3):185–203.

[Hsiao et al., 2010] Hsiao, E., Collet, A., and Hebert, M. (2010). Making specific features less discriminative to improve point-based 3d object recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2653–2660. IEEE.

[Huang et al., 2009] Huang, Y., Liu, Q., and Metaxas, D. (2009). Video object segmentation by hypergraph cut. In *Computer Vision and Pattern Recognition*. IEEE.

[Huertas and Medioni, 1986] Huertas, A. and Medioni, G. (1986). Detection of intensity changes with subpixel accuracy using laplacian-gaussian masks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (5):651–664.

[Jolliffe and MyiLibrary, 2002] Jolliffe, I. and MyiLibrary (2002). *Principal component analysis*, volume 2. Wiley Online Library.

[Kaess et al., 2011] Kaess, M., Johannsson, H., Roberts, R., Ila, V., Leonard, J., and Dellaert, F. (2011). isam2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3281–3288. IEEE.

[Katz and Brock, 2008] Katz, D. and Brock, O. (2008). Manipulating articulated objects with interactive perception. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 272–277. IEEE.

[Katz and Brock, 2011] Katz, D. and Brock, O. (2011). Interactive segmentation of articulated objects in 3d. In *Workshop on Mobile Manipulation at ICRA 2011*.

[Kenney et al., 2010] Kenney, J., Buckley, T., and Brock, O. (2010). Interactive segmentation for manipulation in unstructured environments. In *IEEE International Conference on Robotics and Automation, 2009. ICRA'09.*, pages 1377–1382. IEEE.

[Lazebnik and Raginsky, 2009] Lazebnik, S. and Raginsky, M. (2009). Supervised learning of quantizer codebooks by information loss minimization. *IEEE TPAMI*, 31(7):1294–1309.

[Lazebnik et al., 2006] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition*.

[Li and Fei-Fei, 2010] Li, L. and Fei-Fei, L. (2010). Optimal: Automatic online picture collection via incremental model learning. *International Journal of Computer Vision*, 88:147–168.

[Li and Kleeman, 2009] Li, W. and Kleeman, L. (2009). Interactive learning of visually symmetric objects. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 4751–4756. IEEE.

[Li and Kleeman, 2011] Li, W. and Kleeman, L. (2011). Segmentation and modeling of visually symmetric objects by robot actions. *The International Journal of Robotics Research*, : .

[Lian et al., 2010] Lian, X.-C., Li, Z., Lu, B.-L., and Zhang, L. (2010). Max-margin dictionary learning for multiclass image catagorization. In *European Conference on Computer Vision*.

[Lindeberg, 1994] Lindeberg, T. (1994). Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1):225–270.

[Lippiello et al., 2007] Lippiello, V., Siciliano, B., and Villani, L. (2007). A position-based visual impedance control for robot manipulators. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2068–2073. IEEE.

[Lowe, 2004a] Lowe, D. (2004a). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 1:91–110.

[Lowe, 2004b] Lowe, D. (2004b). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

[Ma and Manjunath, 1997] Ma, W. and Manjunath, B. (1997). Netra: A toolbox for navigating large image databases. In *Image Processing, 1997. Proceedings., International Conference on*, volume 1, pages 568–571. IEEE.

[MacQueen et al., 1967] MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14.

[Mansfield and O'Sullivan, 2011] Mansfield, M. and O'Sullivan, C. (2011). *Understanding physics*. Wiley.

[Maron and Kuhns, 1960] Maron, M. and Kuhns, J. (1960). On relevance, probabilistic indexing and information retrieval. *Journal of the ACM (JACM)*, 7(3):216–244.

[Marr, 1982] Marr, D. (1982). *Vision: A computational investigation into the human representation and processing of visual information*. W. H. Freeman & Company.

[Martin et al., 2004] Martin, D., Fowlkes, C., and Malik, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(5):530–549.

[Massey Jr, 1951] Massey Jr, F. (1951). The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, pages 68–78.

[Matas et al., 2004] Matas, J., Chum, O., Urban, M., and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767.

[Metta and Fitzpatrick, 2003] Metta, G. and Fitzpatrick, P. (2003). Early integration of vision and manipulation. *Adaptive behavior*, 11(2):109–128.

[Minka, 2003] Minka, T. (2003). Estimating a dirichlet distribution. *Annals of Physics*, 2000(8):1–13.

[Mishra, 2010] Mishra, A. (2010). *A Fixation Based Segmentation Framework*. PhD thesis, National University of Singapore.

[Mishra et al., 2009] Mishra, A., Aloimonos, Y., and Fah, C. (2009). Active segmentation with fixation. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 468–475. IEEE.

[Mitchell, 1997] Mitchell, T. (1997). Machine learning. wcb. *Mac Graw Hill*, page 368.

[Nelson et al., 1995] Nelson, B., Morrow, J., and Khosla, P. (1995). Improved force control through visual servoing. In *American Control Conference, 1995. Proceedings of the*, volume 1, pages 380–386. IEEE.

[Newcombe and Davison, 2010] Newcombe, R. and Davison, A. (2010). Live dense reconstruction with a single moving camera. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1498–1505. IEEE.

[Nistér and Stewénius, 2006] Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition*.

[Omrcen et al., 2007] Omrcen, D., Ude, A., Welke, K., Asfour, T., and Dillmann, R. (2007). Sensorimotor processes for learning object representations. In *Humanoid Robots, 2007 7th IEEE-RAS International Conference on*, pages 143–150. IEEE.

[Paletta and Pinz, 2000] Paletta, L. and Pinz, A. (2000). Active object recognition by view integration and reinforcement learning. *Robotics and Autonomous Systems*, 31(1):71–86.

[Paris, 2008] Paris, S. (2008). Edge-preserving smoothing and mean-shift segmentation of video streams. In *European Conference on Computer Vision*, pages 460–473. Springer.

[Paris and Durand, 2007] Paris, S. and Durand, F. (2007). A topological approach to hierarchical segmentation using mean shift. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Pellejero et al., 2004] Pellejero, O., Sag
"ués, C., and Guerrero, J. (2004). Automatic computation of the fundamental matrix from matched lines. *Current Topics in Artificial Intelligence*, :197–206.

[Petkos, 2008] Petkos, G. (2008). Learning dynamics for robot control under varying contexts.

[Philbin et al., 2010] Philbin, J., Sivic, J., and Zisserman, Z. (2010). Geometric latent dirichlet allocation on a matching graph for large-scale image datasets. *International Journal of Computer Vision*.

[Ramanathan et al., 2010] Ramanathan, S., Katti, H., Sebe, N., Kankanhalli, M., and Chua, T. (2010). An eye fixation database for saliency detection in images. In *Computer Vision–ECCV 2010*, pages 30–43. Springer.

[Ribas et al., 2010] Ribas, D., Ridao, P., and Neira, J. (2010). *Underwater slam for structured environments using an imaging sonar*, volume 65. Springer Verlag.

[Ross, 2000] Ross, M. (2000). *Exploiting texture-motion duality in optical flow and image segmentation*. PhD thesis, Citeseer.

[Sahinkaya, 2001] Sahinkaya, M. (2001). Input shaping for vibration-free positioning of flexible systems. *Proceedings of the Institution of mechanical engineers, part I: Journal of Systems and Control Engineering*, 215(5):467–481.

[Sahinkaya, 2004] Sahinkaya, M. (2004). Inverse dynamic analysis of multiphysics systems. *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, 218(1):13–26.

[Santos and Ferreira, 2009] Santos, C. and Ferreira, M. (2009). Timed trajectory generation using dynamical systems: Application to a puma arm. *Robotics and Autonomous Systems*, 57(2):182–193.

[Santos and Lima, 1993] Santos, J. and Lima, P. (1993). Multi-robot cooperative object localization–decentralized bayesian approach. *Robot Soccer World Cup XIII, Lecture Notes in Artificial Intelligence*, 5949:332–343.

[Schaal et al., 2003] Schaal, S., Ijspeert, A., and Billard, A. (2003). Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences*, 358(1431):537–547.

[Schaeffer, 2007] Schaeffer, S. (2007). Graph clustering. *Computer Science Review*, 1(1):27–64.

[Schölkopf et al., 1997] Schölkopf, B., Smola, A., and Müller, K. (1997). Kernel principal component analysis. *Artificial Neural NetworksICANN'97*, pages 583–588.

[Scott, 1992] Scott, D. (1992). *Multivariate density estimation*, volume 139. Wiley Online Library.

[Sinha et al., 2006] Sinha, S., Frahm, J., Pollefeys, M., and Genc, Y. (2006). Gpu-based video feature tracking and matching. In *EDGE, Workshop on Edge Computing Using New Commodity Architectures*, volume 278. Citeseer.

[Sivic and Russell, 2005] Sivic, J. and Russell, B. (2005). Discovering object categories in image collections. In *International Conference on Computer Vision*.

[Sivic et al., 2005] Sivic, J., Russell, B., Efros, A., Zisserman, A., and Freeman, W. (2005). Discovering object categories in image collections.

[Soatto, 2009] Soatto, S. (2009). Actionable information in vision. In *Proceedings of the International Conference on Computer Vision*.

[Spong and Vidyasagar, 2008] Spong, M. and Vidyasagar, M. (2008). *Robot dynamics and control*. Wiley-India.

[Stein et al., 2007] Stein, A., Hoiem, D., and Hebert, M. (2007). Learning to find object boundaries using motion cues. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE.

[Strasdat et al., 2010] Strasdat, H., Montiel, J., and Davison, A. (2010). Scale drift-aware large scale monocular slam. In *Proceedings of Robotics: Science and Systems (RSS)*, volume 2, page 5.

[Su et al., 2009] Su, H., Sun, M., Fei-Fei, L., and Savarese, S. (2009). Learning a dense multi-view representation fordetection, viewpoint classication and synthesis of object categories. In *International Conference on Computer Vision*.

[Sudderth et al., 2005] Sudderth, E., Torralba, A., Freeman, W., and Willsky, A. (2005). Describing visual scenes using transformed dirichlet processes. In *Neural Information Processing*.

[Sugar and James, 2003] Sugar, C. and James, G. (2003). Finding the number of clusters in a dataset. *Journal of the American Statistical Association*, 98(463):750–763.

[Thomas et al., 2006] Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., and Gool, L. V. (2006). Towards multi-view object class detection. In *Computer Vision and Pattern Recognition*.

[Thrun et al., 2006] Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Winning the darpa grand challenge. *Journal of Field Robotics*. accepted for publication.

[Torr, 1998] Torr, P. (1998). Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 356(1740):1321.

[Torr and Zisserman, 1998] Torr, P. and Zisserman, A. (1998). Concerning bayesian motion segmentation, model averaging, matching and the trifocal tensor. In *European Conference on Computer Vision (ECCV)*, pages 511–527. Springer.

[Tuytelaars et al., 2010] Tuytelaars, T., Lampert, C., Blaschko, M., and Buntine, W. (2010). Unsupervised object discovery: A comparison. *International journal of computer vision*, 88(2):284–302.

[Tyler, 2008] Tyler, D. (2008). Robust statistics: Theory and methods. *Journal of the American Statistical Association*, 103(482):888–889.

[Ude et al., 2008] Ude, A., Omrcen, D., and Cheng, G. (2008). Making object learning and recognition an active process. *IJ Humanoid Robotics*, 5(2):267–286.

[Vazquez-Reina et al., 2010] Vazquez-Reina, A., Avidan, S., Pfister, H., and Miller, E. (2010). Multiple hypothesis video segmentation from superpixel

flows. In *European Conference on Computer Vision (ECCV)*, pages 268–281. Springer.

[Vedaldi and Fulkerson, 2008] Vedaldi, A. and Fulkerson, B. (2008). Vlfeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/.

[Vernon et al., 2007] Vernon, D., Metta, G., and Sandini, G. (2007). A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents. *Evolutionary Computation, IEEE Transactions on*, 11(2):151–180.

[Weber et al., 2000] Weber, M., Welling, M., and Perona, P. (2000). Unsupervised learning of models for recognition. In *European Conference on Computer Vision*, pages 18–32.

[Weiss and Adelson, 1996] Weiss, Y. and Adelson, E. (1996). A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *cvpr*, page 321. Published by the IEEE Computer Society.

[Welke et al., 2010] Welke, K., Issac, J., Schiebener, D., Asfour, T., and Dillmann, R. (2010). Autonomous acquisition of visual multi-view object representations for object recognition on a humanoid robot. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2012–2019. IEEE.

[Williamson, 2009] Williamson, R. (2009). *Interactive perception for cluttered environments*. PhD thesis, Clemson University.

[Willimon et al., 2010] Willimon, B., Birchfield, S., and Walker, I. (2010). Rigid and non-rigid classification using interactive perception. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1728–1733. IEEE.

[Winn et al., 2005] Winn, J., Criminise, A., and Minka, T. (2005). Object categorization by learned universal visual dictionary. In *International Conference on Computer Vision*.

[Wood et al., 2005] Wood, S., Wood, E., Boyd, D., Pearson/Allyn, and Bacon (2005). *Mastering the world of psychology*. Pearson A and B.

[Wu et al., 2008] Wu, H., Wang, Y., and Cheng, X. (2008). Incremental probabilistic latent semantic analysis for automatic question recommendation. In *ACM conference on Recommender systems*, pages 99–106.

[X. Song et al., 2005] X. Song, X., Lin, C.-Y., Tseng, B., M.-T., and Sun (2005). Modeling and predicting personal information dissemination behavior. In *ACM SIGKDD*.

[Yuan et al., 2007] Yuan, J., Wu, Y., and Yang, M. (2007). Discovery of collocation patterns: from visual words to visual phrases. In *Computer Vision and Pattern Recognition*.

[Zhao et al., 2010] Zhao, B., Fei-Fei, L., and Xing, E. P. (2010). Image segmentation with topic random field. In Daniilidis, K., Maragos, P., and Paragios, N., editors, *Computer Vision – ECCV 2010*, volume 6315 of *Lecture Notes in Computer Science*, pages 785–798. Springer.

[Zhao and Badler, 1994] Zhao, J. and Badler, N. (1994). Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics (TOG)*, 13(4):313–336.